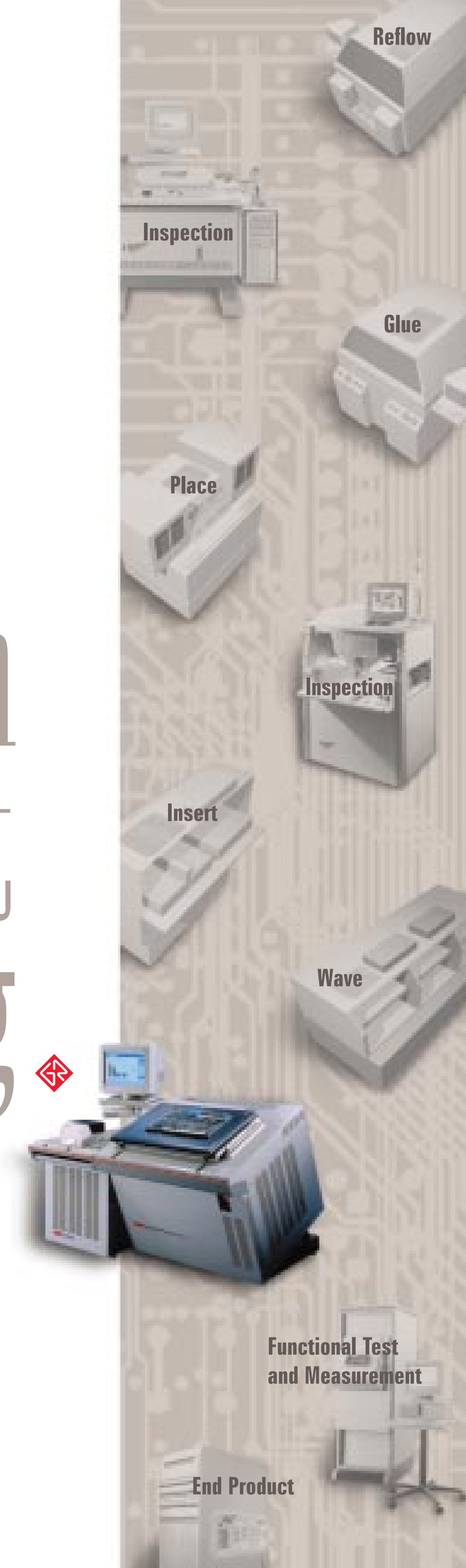
Managing Your Manufacturing Process

Focus on In-Circuit Testing



GenRad®

The Technology of Knowledge

Managing Your
Manufacturing Process

Focus on In-Circuit Testing

(Fourth Edition) ©GenRad, Inc. 1997. All rights reserved. Westford, Massachusetts, U.S.A. 01886-0033. November, 1997.

Foreword

This book addresses a wide audience of professionals: development engineers, quality engineers, test engineers, manufacturing supervisors, project managers, and many others. Its goal is to help the reader understand and deal with the issues related to the manufacturing and testing of printed circuit boards (PCBs). This knowledge can be helpful in many ways:

- Increasing profits and productivity of PCB manufacturing facilities
- Reducing production and labor costs
- Improving time-to-market

To accomplish these goals, we will briefly describe how PCBs are assembled, how the assembly process may inadvertently introduce various types of defects in the finished board, and how you can identify and eliminate these defects. The emphasis in this book is on the role that in-circuit test and inspection systems can play in this process, and on the test strategies that can be used to attain these goals.

The book is styled for easy reading. Emphasis is placed on general concepts and overall flow rather than on specific details of operation. In each chapter you will find the following:

Chapter 1 - **The PCB Manufacturing Process** briefly describes how PCBs are manufactured and the types of defects that can be introduced during the process.

Chapter 2 - **Printed Circuit Board Test and Inspection** examines the reasons for inspecting and testing PCBs, the various methods used, and introduces in-circuit testers.

Chapter 3 - **Techniques for In-Circuit Test and Inspection** presents the critical concepts of accessing the circuit nodes via a bed-of-nails fixture and isolating components by guarding (analog) and backdriving (digital) techniques.

Chapter 4 - **A Look at an In-Circuit Tester** describes the hardware and software components of an in-circuit tester and the functions that these components perform.

Chapter 5 - **Using the Tester** outlines the step-by-step procedure by which test sets are developed.

Chapter 6 - **Techniques for Improving Fault Coverage** presents additional test techniques that can be used to develop the right test strategy for particular applications.

For convenience, a **Glossary** of terms associated with in-circuit testing is included at the back of this document.

Contents

Chapter 1 - The Printed Circuit Board Manufacturing Process 6

Chapter 2 - Printed Circuit Board Test and Inspection 14

Chapter 3 - Techniques for In-Circuit Test and Inspection 19

Chapter 4 - A Look at an In-Circuit Tester 48

Chapter 5 - Using the Tester 60

Chapter 6 - Techniques for Improving Fault Coverage 78

Glossary 94

BusBust is a registered trademark of GenRad, Inc. Panel-Test, Opens Xpress, TestFlo, Xpress Model and Scratchprobing are trademarks of GenRad, Inc.

Chapter One

The Printed Circuit Board Manufacturing Process

Chapter One

The Printed Circuit Board Manufacturing Process

An overview

Before we investigate how printed circuit boards (PCBs) are tested, we'll first review the process which is used to manufacture them. We'll also familiarize ourselves with the types of defects and faults that the PCB manufacturing process can introduce along the way. There are many variations depending on the specific process and manufacturing equipment available.

How are PCBs made?

Here are the major steps in the process:

1. The PCB is designed using Computer-Aided Design (CAD) software.
2. Manufacturing uses the information in the CAD database to instruct the machines used to build and assemble the board.
3. The bareboard is fabricated.
4. Solder screen printing may be done.
5. Components are placed on the board and soldered.
6. The PCB is cleaned.
7. The PCB is inspected for process faults.

Each of these steps is described in more detail in this chapter.

Step 1 - Designing the CAD database for a PCB

Today's PCBs can be quite complex and PCB designers need all the help they can get to develop these boards efficiently. That's why Computer-Aided Design (CAD) software plays such an important role in PCB design.

The PCB designer starts by gathering all necessary circuit information: schematic diagrams, component descriptions, parts lists, and any other helpful documentation. With this circuit information in hand, the PCB designer then uses CAD software to define the physical dimensions and layout of the board, and specifies the location of all its components and interconnections.

One big advantage of using CAD software is that it allows PCB designers to simulate their design on a computer and modify the design until it meets their requirements. It's a trial and error approach but with the advantage of not having to build and rebuild the board over and over again. This saves both time and money.

Once the design is complete, the CAD software produces a master drawing showing dimensions and grid locations including the arrangement of conductors and nonconductive patterns (the size, type, and location of vias, etc.). CAD software also creates a database that contains the name of each component, its part number, orientation, exact location on the board, and much more.

The designer can also use the CAD simulation to check the operation of the board design and verify that appropriate considerations have been given to the manufacturing and testing of the board.

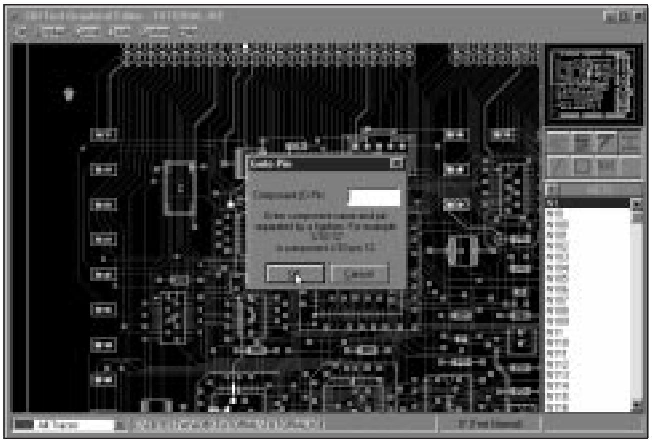
Step 2 - Setting up the manufacturing process for a PCB

Manufacturing uses the CAD design data to build and assemble the PCBs. To do this successfully, manufacturing must have clear and complete documentation, including the following items:

- Bill of materials
- Fabrication and assembly details
- Gerber data
- Approved vendor lists
- Special assembly instructions

Manufacturing must also have access to a complete and accurate CAD database for the PCB. This CAD data is especially useful for:

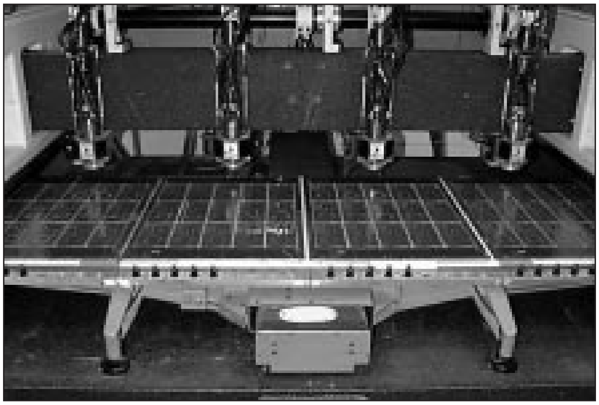
- Developing machine tooling
- Fabricating the solder paste stencil (which we'll learn about later)
- Programming automated assembly systems
- Preparing test fixtures



Step 3 - Fabricating the bareboard

The first step in the manufacturing process is to fabricate the bare PCB. As the name implies, the bareboard contains no components, but it does contain all the holes and electrical 'tracks' that will be needed later to mount and connect the components. The following figure shows a routing machine processing several multi-up panels at once. The router drills plated and in-plated holes, routes breakaways, and cuts panels into individual boards.

NOTE: The lands and pads are the portion of the PCB used for attaching surface mount components to the board and for making the electrical connections. Lands and pads must be carefully designed to ensure good component solder processing.



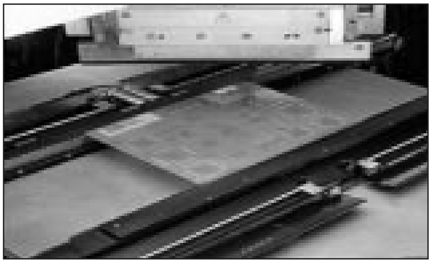
Step 4 - Solder/Screen Printing

A stenciling process is sometimes used to apply solder paste to boards that contain surface mount assemblies. The solder is applied in the form of a paste consisting of tiny solder beads in a flux base. The stencil is a thin stainless-steel sheet with a circuit pattern cut and etched into the material. The stenciling process works as follows:

1. The bareboard is held in place by a vacuum or by some mechanical means.
2. The stencil is placed on the board and aligned so that the stencil openings overlay the component lands which require soldering.
3. A bead of solder paste or adhesive is pushed through the stencil openings onto the lands by a rubber or metal blade moving slowly across the stencil. The solder paste is used for electrical connections. The non-conductive adhesive is used to hold the surface mount devices in place during wave soldering in particular applications.
4. The stencil is lifted away leaving solder paste or adhesive deposits at the appropriate locations on the board.

Defects:

- Skipped pads
- Paste slump
- Insufficient solder
- Solder balling
- Smudged print
- Uneven print
- Misaligned dispense
- Misaligned print



Step 5 - Placing the components on the board

PCBs can contain one or both of two major types of components: through-hole components and surface mount devices (SMDs). As shown later in this chapter, the manufacturing process may differ slightly depending on which of these two types of components are used on the PCB.

After the components are placed on the board, they are soldered. Two major soldering techniques are used in PCB manufacturing: wave soldering and reflow soldering. These soldering techniques are described later in this chapter.

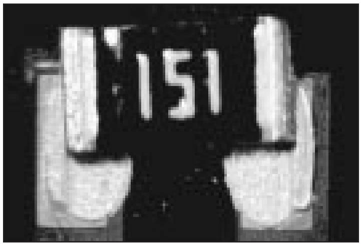
NOTE: Some components that might be damaged by the extreme temperature conditions during soldering or cleaning, are assembled and soldered manually by an operator after the soldering and cleaning processes are complete.

A. Placing Surface Mount Devices

Surface Mount Devices (SMDs) - SMDs are held in feeders and placed on the board by a mounting mechanism with a vacuum tip. The mounting mechanism finds the correct location, places the component, and exerts a precise force on the component causing it to penetrate the solder paste or glue.

Defects:

- Cracked component
- Incorrect placement
- Bent or lifted lead
- Misplaced component

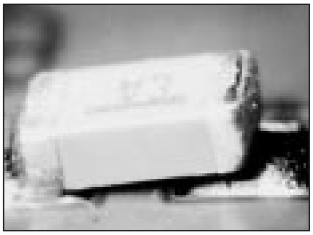


The mounting mechanisms that are used for placing components on the board are controlled by program instructions developed from the CAD data. Each component requires a program instruction that specifies the component name, the feeder where it is located, the component's X and Y coordinates, its orientation, and sometimes its height.

Reflow Soldering - Reflow soldering is a bonding method in which a solder paste is applied to those surfaces of a PCB where components are being attached to the board. The paste, previously applied by the stencil process, is then heated so that it melts and forms a solder joint.

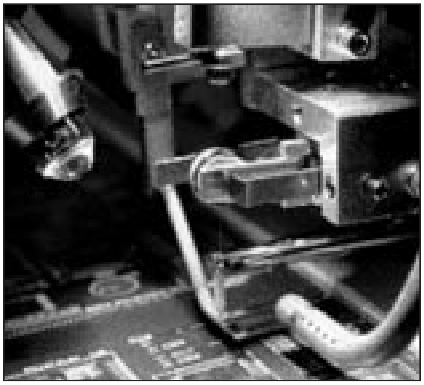
Defects:

- Fractured solder joints
- Poor wetting/cold solder
- Component lift
- Poor solderability
- Solder beading
- Solder short
- Component cracking



NOTE: Wave soldering, which is described on the next page, is common for lower profile components such as diodes and transistor packages that are installed on the bottom side of the board. Wave soldering is possible for some SMD components, but reflow soldering is preferred.

To wave solder SMDs, attachment adhesives are used to hold the devices in place while the PCB is turned over and passed through the solder wave. The entire device is submerged in the wave, and solder adheres to the solderable surface only.



B. Placing Axial and Radial Components

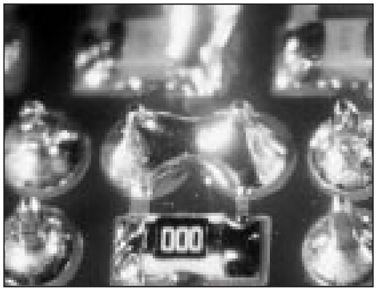
Through-hole components (auto insertion) - These components, which include axial-lead components, radial-lead components, and dual-inline-packages (DIPs), are placed on the board using insertion machinery. After the component is inserted, its leads are clinched to meet design specifications.

Through-hole components (manual insertion) - Components that cannot be inserted with machinery, such as relays, DC/DC converters, etc, are inserted by hand along sequential lines.

Wave soldering - The board is passed over a continuous wave of molten solder, which solders all the joints simultaneously. The PCB is kept in contact with the solder long enough for the leads of the components to heat up and form a stable bond with the solder.

Defects:

- Solder balling
- Insufficient top solder
- Solder skip
- Excess solder
- Poor wetting
- Solder short
- Blow hole
- Incomplete joint
- Solder webbing



The process flow may vary

The sequence of steps in the manufacturing process of PCBs with surface mounted devices may vary depending on whether the board contains only SMDs or both SMDs and through-hole devices. The following chart summarizes the major differences in the process flow.

SMDs only

- Apply solder paste
- Place SMDs on board
- Reflow solder
- Inspection

Through-hole components on top, SMDs on the bottom

- Insert and clinch through-hole components
- Turn the board over
- Apply adhesive
- Place SMDs
- Turn board over
- Solder entire assembly
- Inspection

Through-hole components on top, SMDs on top and bottom

- Apply solder paste to top of board
- Place SMDs on top
- Reflow solder
- Insert and clinch through-hole components
- Turn board over
- Apply adhesive to bottom side
- Place SMDs on bottom
- Turn board over again
- Wave solder all components
- Inspection

Step 6 - Cleaning the PCB

Regardless of how much care is taken in manufacturing, a PCB is still exposed to many sources of contamination. For example:

- The bare board may be exposed to contamination during packaging, storage, shipping, receiving, and assembly.
- Boards may also be contaminated when handled and moved from station to station for component insertion and soldering.
- The flux used in soldering the components may contain residues that can conduct electrical current or that can act as insulators. These residues can cause shorts, corrode solder joints, or inhibit current flow across connectors.

A cleaning process is necessary to remove the flux and other possible contamination prior to test and inspection. Some companies use no-clean pastes and fluxes in which the remaining residues are inert. Sometimes, these types of fluxes create test problems, since the remaining residue can encapsulate and isolate the test points. A properly selected flux type will not create this problem.

Step 7 - Inspecting/testing the board

The PCB is always inspected to some degree during the assembly process. For example, after the components are placed on the board, an inspector will look for gross defects such as smeared solder paste or missing components. After components are soldered, the inspector will visually check the quality of reflowed solder joints.

But, you can't detect all possible manufacturing faults with just visual inspections.

Also, visual inspections are not practical for collecting and evaluating large amounts of data on how defects are distributed during the manufacturing process. This type of data can be very useful in reducing future defect levels. Automatic test and inspection systems are more suited to that task.

Typically, there are always some defects in circuits built on a production line. Therefore, the inspection and testing phase of PCB manufacturing is essential. In Chapter 2, we'll examine just how important it is to inspect and test the PCBs. We'll also investigate some of the methods available for inspecting and testing. Then in Chapter 3, we'll start to focus on one of those methods, called ***in-circuit testing.***

Chapter Two

Printed Circuit Board Test and Inspection

Chapter Two

Printed Circuit Board Test and Inspection

Why should I bother to test?

The answer is purely economics. If a product is not tested, yields will not be acceptable, manufacturing will not meet production needs, defective products will reach customers, and inventory costs will increase. Also, it will be more difficult to discover and solve process problems which affect repair and rework costs. In short, it is simply more expensive not to test.

Consider the cost of finding and fixing defects.

Here is an indication of the relative cost of finding and fixing defects at various stages in the product cycle:

Manufacturing process

To find a defective component at incoming inspection might cost about 1X

On board

If a defect slips past this stage, the cost of finding a bad part once it has been mounted on the PCB goes up to about 10X

Final assembly

The cost of finding and replacing that same part once the PCB is passed on to final (system) assembly now becomes 100X

In the field

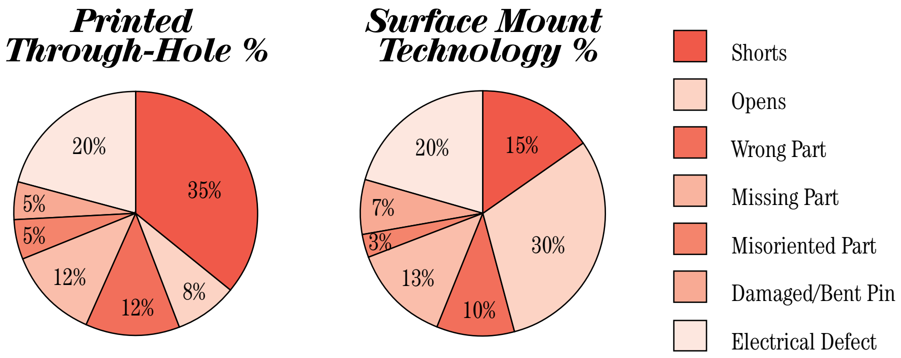
Finally, if the system finds its way into the field with that defective part and appears as a warranty repair, the expected cost approaches 1000X

What types of defects can the manufacturing process introduce?

As we’ve seen in Chapter 1, defects can be inadvertently introduced at any stage of the PCB manufacturing process. Here are some of the more common faults:

- **Shorts -** Caused by excessive solder and are the result of a poor soldering process or a poor board design.
- **Opens -** Caused by insufficient solder.
- **Missing components -** Could occur at one of the component placement stages due to a feeder problem.
- **Misoriented components -** Typically occurs when components are improperly placed in the feeder.
- **Wrong components -** Could occur if the wrong component was loaded into the feeder or if the wrong feeder was loaded.
- **Damaged components -** Some components can be damaged during the manufacturing process.
- **Tombstoning -** A picturesque term describing a chip that has worked its way into a vertical position leaving only one end soldered. Tombstoning is caused by soldering defects.
- **Poorwetting -** Caused by board contamination, solderability problems, or a poor soldering process.

Typical fault distribution



NOTE: Even if the bare PCB had no defects and you tested every component before connecting it to the board, you could still introduce problems while assembling the board.

So, what do I need to test and inspect PCBs? A few test instruments and some extra time should do the job.

Maybe, but don't underestimate the job.

First, you have to understand how all the circuits on the board work before you can figure out how to test them. Once you've gotten over this hurdle, you have to write test procedures specifying exactly how the board will be tested. Remember, you want to test all the circuits; otherwise you can't be sure the board will work properly under all normal operating conditions.

Once you've written the test procedures, you have to assemble all the necessary test equipment: oscilloscopes, voltage sources, current sources, meters, and so on.

Manual test and inspection

Now, connect all this equipment to the appropriate points on the PCB. Some of those points may be hard to reach. If you have many boards to test, you may want to build a fixture to simplify connecting the test leads.

Finally, turn on the equipment, apply known inputs to the circuit, check the outputs and determine if the circuit is operating properly. And, if the circuit is not operating properly, find the defective part and replace it.

To make the process even more difficult, PCBs are becoming more compact and complex and components are becoming more and more difficult to access.

That could be quite a bit of work...

Actually, there's more to it than that.

Now, you have to turn off the equipment, re-connect the test leads to other points on the board, turn the equipment back on, drive some new inputs, check some new outputs, etc. You get the idea.

What if the PCB has hundreds or even thousands of circuit connections (nodes)?

And how will you test the more complex components like microprocessors and other VLSI (Very Large-Scale Integration) chips?

And don't forget that all these components are interconnected on the PCB. So, if an output isn't what it should be, how will you know which component is causing the failure?

This testing approach could be quite a job and might not be too successful.

A much more efficient approach would be to use a test system with computer-controlled test instruments and the proper software.

So, tell me how I can automate the process.

The term automatic test equipment (ATE) applies to many forms of programmable, computer-controlled test instruments and systems. This equipment is used in many stages of the manufacturing process, including:

- Component manufacturing
- Incoming inspection
- System testing
- Quality assurance
- PCB test and inspection
- Field service

The following table compares various methods of testing PCBs for potential faults. This book concentrates on the ICT (In-Circuit Test) method because it has proven to be the most effective technique for high volume manufacturing of PCBs.

Test Method/ Approach	Cost	Program Development	Test Speed	Faults Detected	Faults Missed
ICT	low	low	very fast	opens, shorts, missing bypass, tolerance, functional	high-speed parametric
MDA	low	low	very fast	opens, shorts, tolerance	missing bypass, function
Vectorless	low	low	fast	opens	shorts, tolerance, function
Manual Vision	high	none	slow	opens, shorts on visible parts	tolerance, function, unviewable connections
X-ray	high	low	slow	solder opens, shorts, alignment, outline	tolerance, function
B-Scan	low	minimal	medium/fast	opens, shorts on boundary cells, internal	tolerance, function, parts w/o boundary scan
Functional	medium	low to very expensive	fast, except for diagnostics	specified test parameters	component parametrics not evident in the circuit

ICT - In-Circuit Testers. You'll learn a lot about them in this book.

MDA - Manufacturing Defect Analyzers perform a subset of the tests performed by in-circuit testers. Checks for shorts and opens, and performs some analog tests. Uses a test fixture to access circuit nodes.

Vectorless - Uses analog techniques to check for opens on component pins. Sometimes requires special fixture hardware.

Manual Vision - Manual visual inspection.

X-ray - Uses X-ray equipment to check solder joints.

B-Scan - Boundary scan is a technique that requires the device-under-test to contain circuitry that facilitates testing.

Functional - Functional tests are used to verify that the circuit will work when installed in a system.

This book concentrates on one specific category of automatic test equipment: **IN-CIRCUIT TESTERS**

What is an in-circuit tester?

It's a tester that tests each component on a PCB, one at a time. And it does this while the component is "in-circuit," that is, while it's connected to other components on the board. Since each component is tested separately, pinpointing the cause of a failure is a relatively easy job for an in-circuit tester.

The in-circuit tester performs all the manual procedures that were described on page four, namely:

- Writes test procedures
- Produces fixture wiring reports
- Connects the test equipment to the board
- Turns the equipment on and sets it up
- Applies known input signals and checks outputs
- Determines if the circuit is operating properly
- Locates defective components when the circuit is faulty; and last but not least,
- Repeats this for all the circuitry on the board

Manufacturing uses in-circuit testers to test fully assembled PCBs. The testers check individual components and interconnections on the board. First, they run connectivity tests to check for shorts and opens, then analog tests to measure values of analog components, digital tests to check the operation of digital ICs (integrated circuits), and hybrid tests to check components that perform both analog and digital functions. As we'll see in the last chapter, in-circuit testers also provide value-added capabilities such as the programming or downloading of unit under test (UUT) firmware.

Some of the important features of an in-circuit tester are listed below. More detailed descriptions of these features will follow in later chapters.

- Identifies and pinpoints board defects immediately
- Detects most manufacturing faults
- Contains a circuit analyzer, test generator, and component libraries
- Provides system software to write and evaluate tests
- Libraries contain model tests for a wide variety of components

The techniques that in-circuit testers use to isolate and test each separate component on the board, one at a time, are described in Chapters 3 and 4.

Chapter Three

Techniques for In-Circuit Test and Inspection

Chapter Three

Techniques for In-Circuit Test and Inspection

An in-circuit tester must be able to ACCESS and ISOLATE

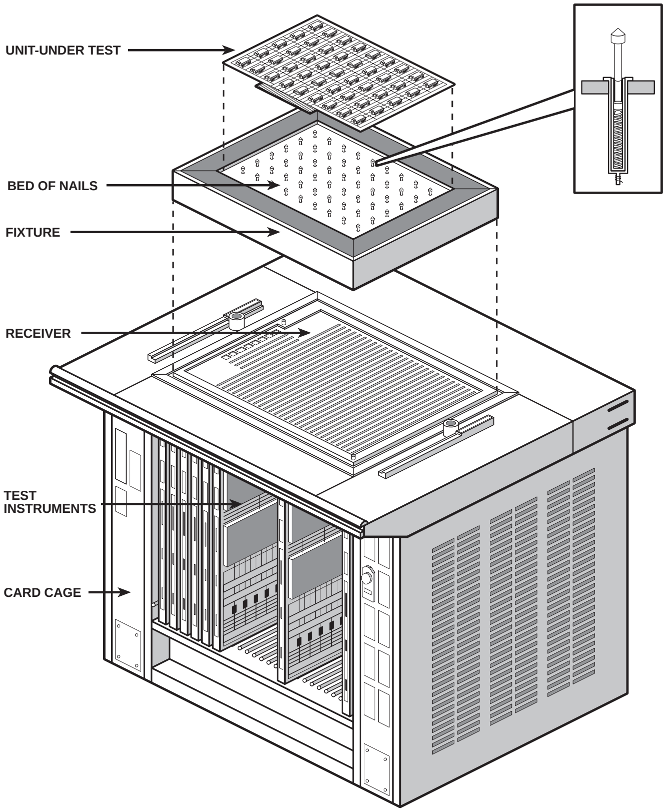
To perform in-circuit testing, the tester must first be able to do two things:

- It must have access to all the components that you want to test. Obviously, to test each component individually, the tester must be able to connect test instruments to each pin of each component.
- Also, it must be able to isolate each component-under-test from surrounding components. Since components are interconnected on the board, special isolation techniques are needed to prevent the component-under-test from being affected by other components.

This chapter introduces the in-circuit tester and describes the basic techniques used by an in-circuit tester to access, isolate, and test components on an assembled PCB. Analog components are addressed first, followed by digital components.

Overview of an in-circuit tester

We'll start with a top-level look at the principal parts of an in-circuit tester. Then we'll examine how those parts are used to access, isolate, and test components on a board.



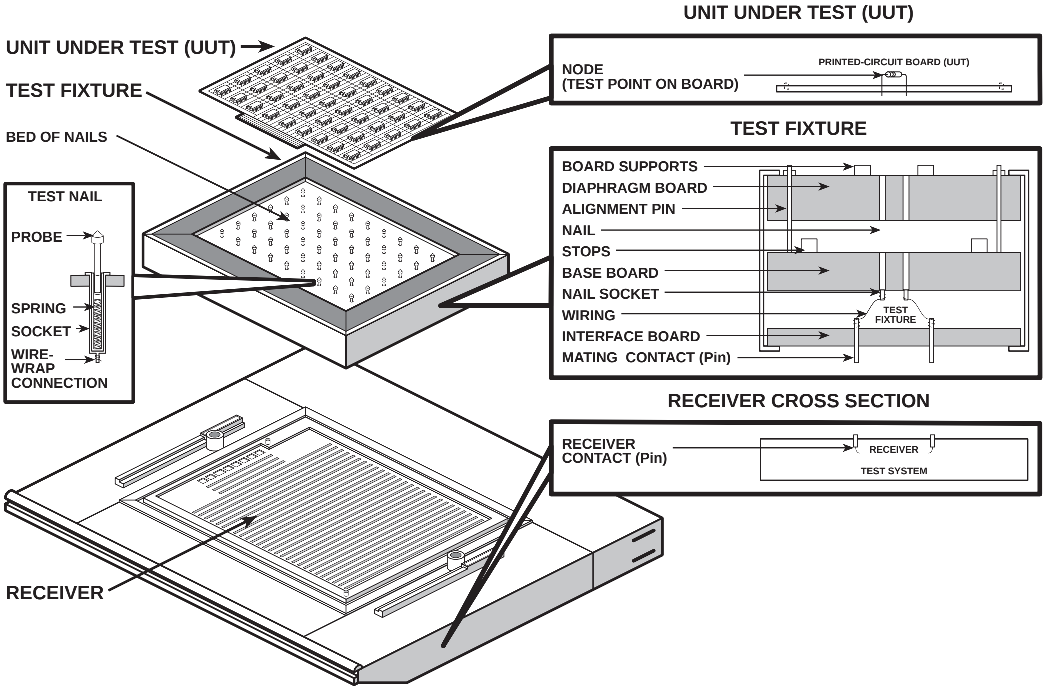
Accessing the circuit nodes with a bed-of-nails

To access the circuit nodes on a PCB, the tester uses a special test fixture, appropriately called a bed-of-nails. The “nails” are small spring-loaded probes that make physical contact with the tracks and component nodes on a board during a test. The nails are mounted in sockets and are connected to the tester by wiring the other end of the sockets to connectors that plug into the system.

Each PCB requires its own unique bed-of-nails test fixture. Each custom-built fixture typically has hundreds of test nails properly located to come in contact with the circuit nodes on the board under test. Chapter 4 provides more details on how these bed-of-nails fixtures are made.

Prior to testing, the board is placed component-side up on the fixture and a vacuum is activated, pulling the board down onto the test nails. In some cases, test nails may need to be placed on both sides of the board to access all component pins. When this is necessary, a special type of fixture is used.

This physical interface between the tester and the fixture is called the receiver.



Some helpful definitions

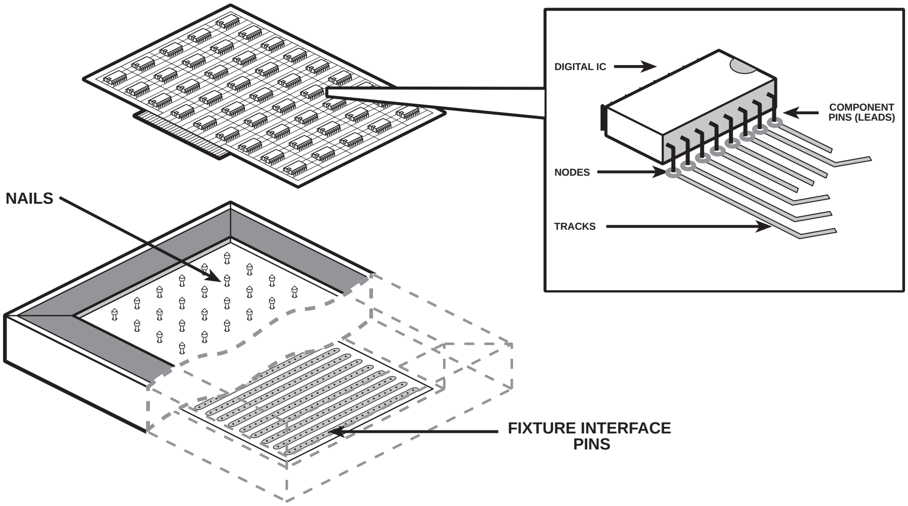
To prevent possible confusion later on, let’s digress a moment and define some of the terms associated with the interface between the tester and the test fixture.

NAIL **(or TEST NAIL)**

Refers to the spring-loaded probes in the test fixture, which come in contact with the PCB under test.

NODE A circuit point (or track) on the PCB itself.

PIN	When the tester is being referred to, the term ‘pins’ (or ‘test pins’) signifies all the test connections available to access the board simultaneously. For example, a tester may be said to have a total of 640 test pins. When a component is being referred to, the term (component) pins signifies the leads attached to that component, for example, pin 4 of IC35.
LEAD	Usually refers to the connection attached to a circuit component (e.g., a capacitor has 2 leads, an IC has 16 leads, etc).



Now, back to accessing circuit nodes

Remember, the whole purpose of the bed-of-nails fixture is to connect the system test instruments to the circuit nodes.

Since all testing is analog, digital, or hybrid (a combination of both), every test nail can be connected to either the analog or digital test instruments, as directed by the test program. Because of this dual capability, these nails are referred to as hybrid nails. The obvious advantage of hybrid nails is that either analog, digital, or hybrid testing can be performed at any circuit node on the board, as the test program dictates.

How do the test instruments access the test nails?

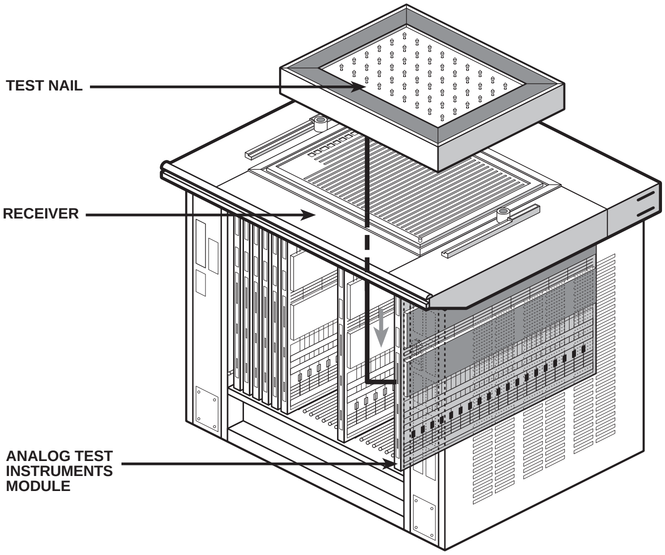
OK. Let’s get acquainted with the test instruments and how they are routed to the circuit nodes on the board. We’ll start with the analog test instruments, which include the following:

- DC current source
- DC voltmeter
- AC voltage source
- Waveform generator
- DC voltage source
- DC ammeter (current meter)
- AC voltmeter

A tester may have additional analog test instruments that expand the capabilities of these standard instruments, but we’ll look at those later in this book.

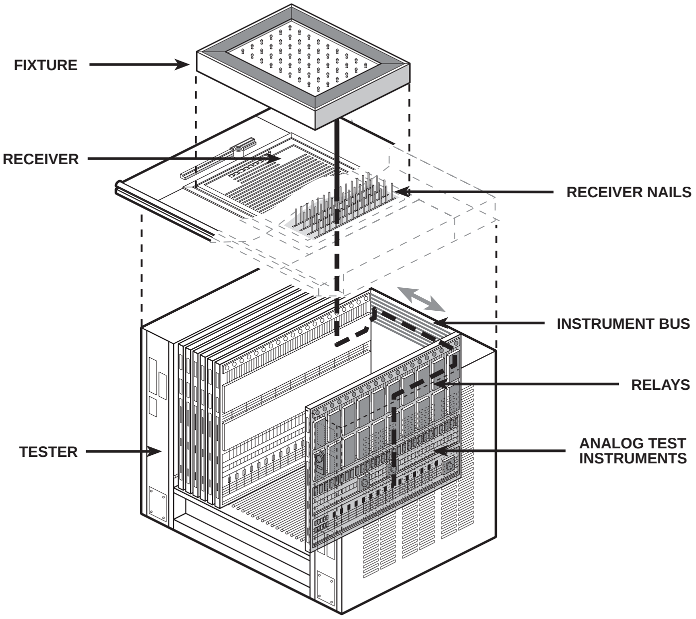
Now, the question is, how can the tester connect any one of these instruments to any one of hundreds of circuit nodes on the board?

It's all done under computer control and directed by a test program uniquely developed by the tester for the specific board or UUT.

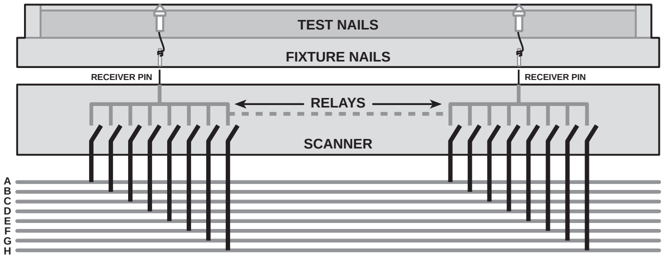


Connecting the analog test instruments to the nodes

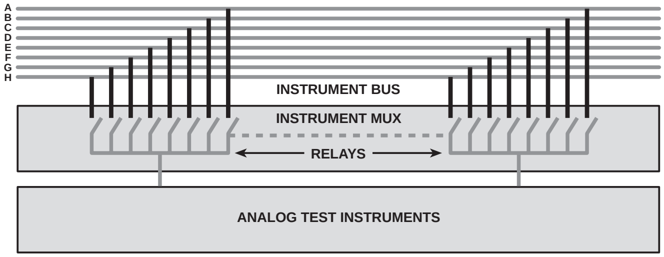
The analog test instruments are connected to the test nails by software-controlled relays, i.e., relays that can be opened or closed by program instructions. These relays route the test instruments and the test nails to a common set of eight lines (or channels) called an instrument bus. The eight channels in the instrument bus are identified by the letters A through H (as illustrated on the next page).



The tester has two sets of relays. One set of relays, called the SCANNER, connects any test nail to any of the eight instrument bus channels (A through H).



The other set of relays, called the INSTRUMENT MULTIPLEXER (or MUX), connects these same eight channels to any analog instrument.



So, by controlling both the MUX and the SCANNER relays, the software can route any analog instrument through the instrument bus to any test nail.

NOTE: Multiplexing is a way of sharing high quality test instruments among many test pins, making cost effective testing possible. As we'll see later, multiplexing is also used to connect the digital test circuitry to the test nails.

Controlling the MUX and SCANNER

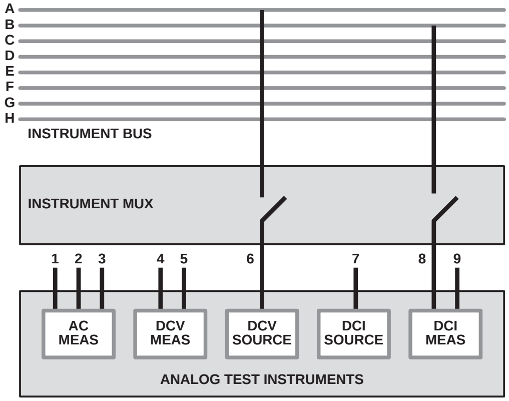
The following example illustrates how the system-developed test program controls the MUX and SCANNER relays, allowing the test instruments to access the test pins.

The program statement:

SET MUX AT (CHA=DCSVHI: CHB=DCMISNS);

closes the relays connecting the high side of the DC voltage source to channel A and the current source input of the DC current meter to channel B.

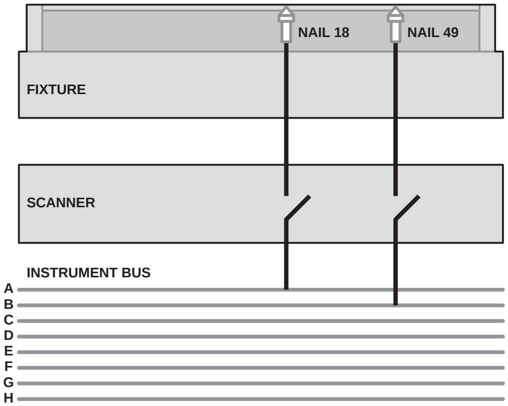
SET MUX AT (CHA=R270.1: CHB=R270.2);



Similarly, the statement:

SET SCAN AT (CHA=R1.1: CHB=R1.2); closes the relays connecting that same MUX channel A to R1.1 and channel B to R1.2.

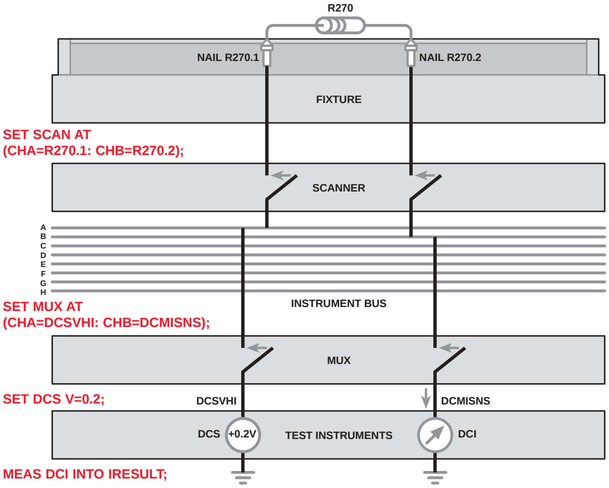
SET MUX AT (CHA=270.1: CHB=R270.2);



Let's see how the instruments are used for in-circuit testing

To continue the example one step further, let's say that the PCB has a resistor connected between nodes R1.1 and R1.2. You could apply a known DC voltage (0.2V) across the resistor, measure the resulting current, and store the results. When testing a UUT, the multiplexer is set once, then all UUT resistors are tested using a MEAS R statement, as illustrated below:

```
SET MUX AT (CHA= DCSVHI: CHB=DCMISNS);
R1: MEAS R   AT (R1.2:R1.1)   V=0.2   I=128M   HI=1.5K   LO=1.4K;
```

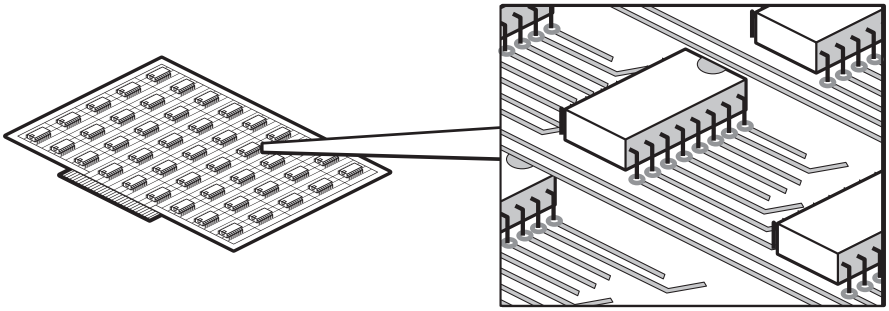


This approach speeds up the testing process.

So, that's how the analog test instruments access the nodes. Now, how does the tester isolate and test components?

The isolation techniques used by the in-circuit tester differ depending on whether the component being tested is an analog component (resistor, capacitor, diode, etc.) or digital device (microprocessor, memory chips, bus driver, etc.).

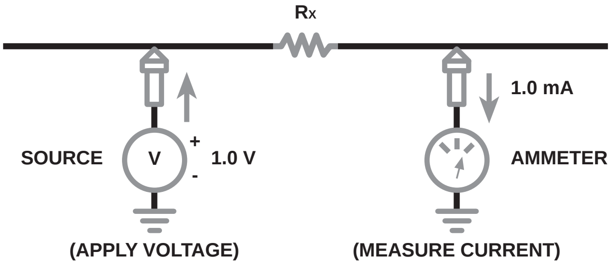
We'll start with analog components, and examine why they must be isolated from the rest of the circuit before they can be tested.



How do you isolate and test an analog component in a circuit?

Suppose you wanted to check the value of a free-standing analog component, one that is not connected in a circuit. You could apply a known input, measure the resulting output, and compute the value of the component by using mathematical formulas that apply to that component.

For example, to test a resistor, you could apply a known voltage across the resistor, measure the resulting current, and calculate its resistance by dividing the value of the applied voltage by the value of the resulting current (Ohm's Law: R=V/I).



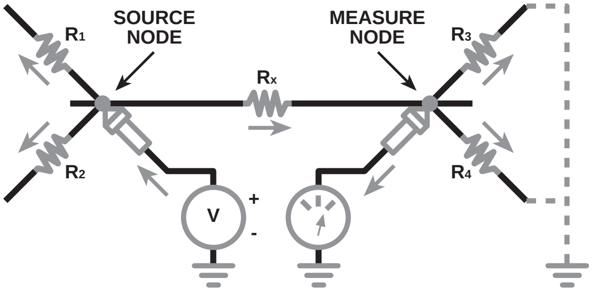
If the applied voltage were 1.0 V and the measured current were 1.0 mA, the calculated resistor value would be

$R_X = V/I = 1.0/0.001 = 1000$ ohms

This is called a 2-terminal measurement.

What happens when that resistor is connected in a circuit?

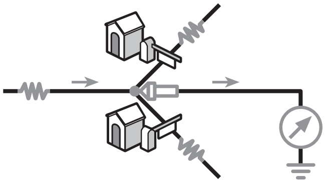
There would very likely be some shunt paths around the resistor (R_X), which would divert some of that resistor current from flowing into the ammeter. Resistors R_1, R_2, R_3 and R_4 represent these shunt paths.



Instead of flowing directly to ground through the ammeter, the total current through R_X would now split up and flow through R_3 and R_4, as well as through the ammeter.

Depending on the resistance values of these shunt paths, the ammeter reading could be affected significantly and, therefore, the calculated value of R_X could be significantly in error.

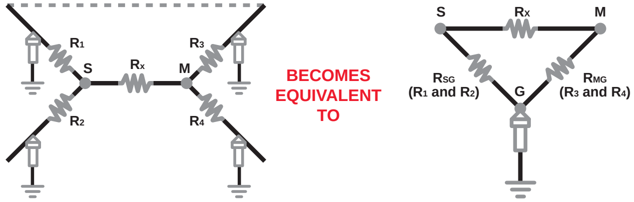
How can the tester "guard" against these unwanted shunt currents?



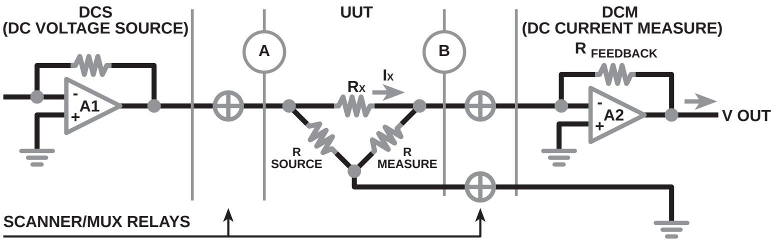
By using a technique known as guarding, which is the key to analog in-circuit testing. Guarding stops, or at least reduces significantly, the current flow through the shunt paths connected to the measure node.

Guarding is accomplished by:

- Temporarily connecting all shunt paths around the component-under-test to ground (the guard voltage), and



- Using an operational amplifier (op amp) in the ammeter circuit.



This is called a 3-terminal guarded measurement.

How does this help?

First, let's consider a very important feature of the op amp circuit.

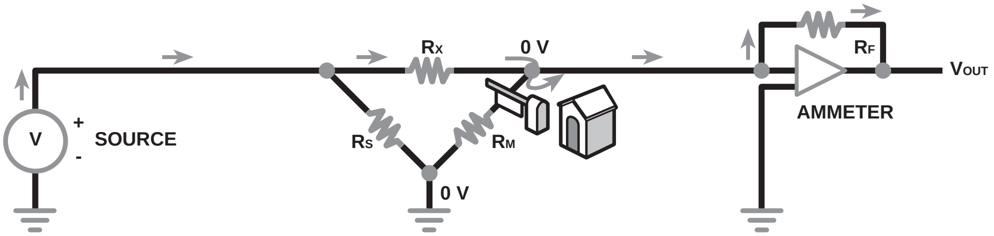
> Because of the feedback resistor (R_F), the - input to the amplifier is forced to be very nearly equal to the + input. Since the + input is tied to ground, the - input is also very close to ground (virtual ground).

Now, back to the guard circuit shown on the previous page. In the ideal case:

> The measure node is at 0V (virtual ground) due to the op amp, and the guard node is also at 0V because it is connected to ground by the tester.

Since there's no voltage across the measure and guard nodes, there's no current flow, right?

Voila! We've temporarily disabled these shunt paths and effectively isolated R_X from the rest of the circuit. All the current through R_X now flows into the ammeter circuit.



The tester knows the value of the feedback resistor (R_F) and can measure the op amp voltage V_{OUT}. Therefore, determining the current flow into the ammeter circuit and calculating R_X is a snap.

$$I_{AMMETER} = \frac{V_{OUT}}{RF}$$

$$R_X = \frac{V_{SOURCE}}{I_{AMMETER}}$$

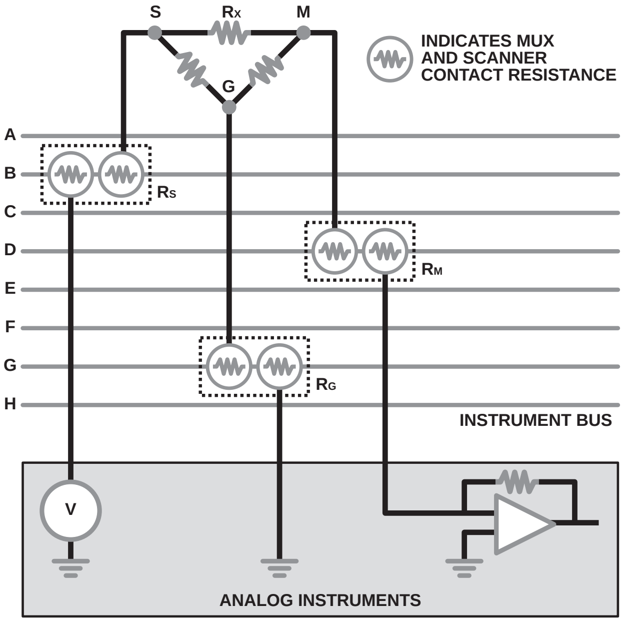
Isn't there still a current flow from the source node to ground?

Yes, there could be.

But, if we ignore such things as cable resistance and switch contact resistance, this current has no effect on the measurement. After all, a known voltage is still being applied across R_X and the ammeter is measuring the resulting current.

In the real-world, however, such mundane things as contact resistance cannot always be ignored and can have an effect on the measurement.

Remember, in the guard circuit just described, we assumed that the guard terminal was at 0V because it was tied directly to ground. In reality, however, the guard terminal is connected to ground through relay contacts in the instrument multiplexer and scanner. Similarly, the source and measure terminals are connected to the PCB through relay contacts.

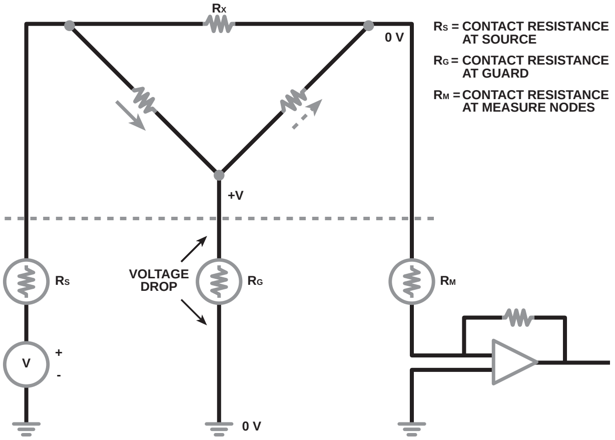


What effect does this current have on the measurement?

Consider the following:

- Any current from the source node to ground flows through this contact resistance.
- Current through this resistance causes a voltage drop, right? So the guard node is not really at 0V.

This means that there is a voltage difference between the measure node and guard node, and some of the shunt current that we tried to guard against is still trickling through.

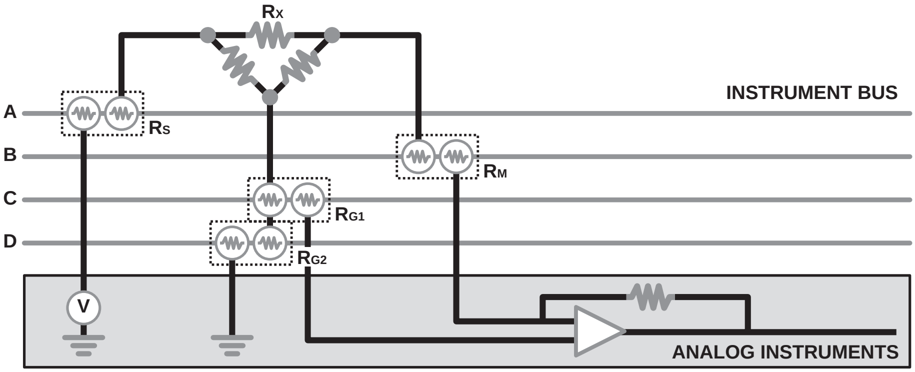

RX
0 V
RS = CONTACT RESISTANCE AT SOURCE
RG = CONTACT RESISTANCE AT GUARD
RM = CONTACT RESISTANCE AT MEASURE NODES
+V
RS
VOLTAGE DROP
RG
RM
V
+
-
0 V

But, we can minimize that problem

By using a fourth terminal (in addition to the source, measure and guard terminals), this source of error can be greatly reduced.

Rather than connecting the ammeter reference (the + input of the op amp) directly to ground, this fourth terminal connects that reference closer to the actual guard nodes on the board.

With this set-up, any error voltage developed across contact resistance R_{G2} now appears at both the guard and measure nodes. This reduces the voltage difference between these 2 nodes and thus reduces that bothersome shunt current at the measure node.



NOTES:

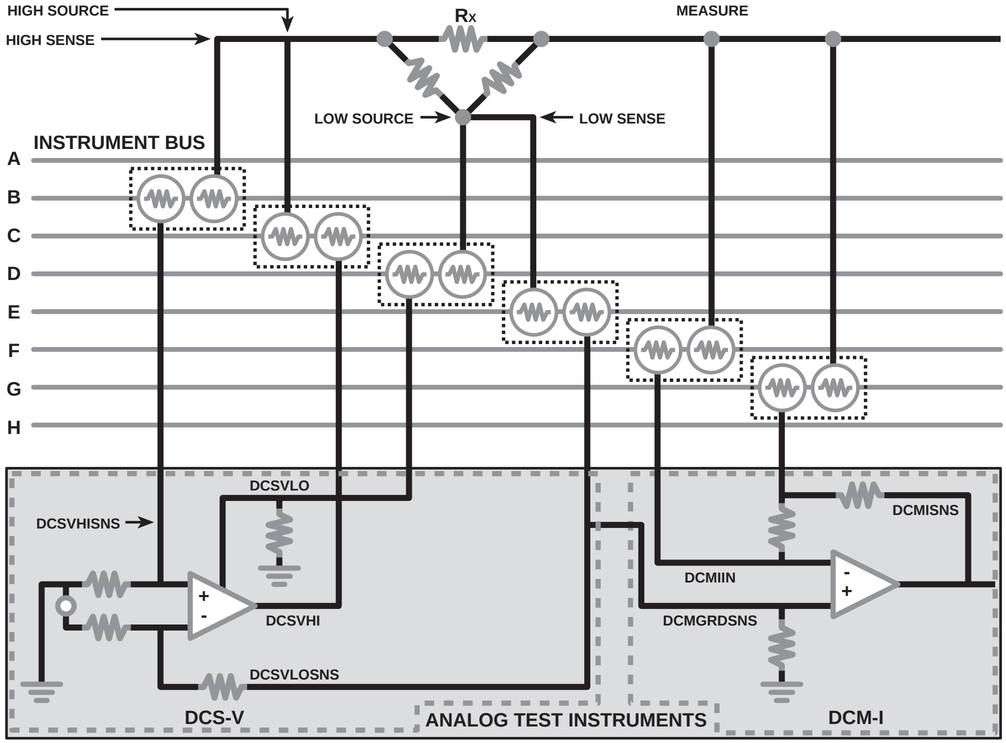
1. Since no current flows into the op amp, no voltage is developed across contact resistance R_{G1}.
2. Except when making low-resistance measurements, the contact resistance at the source and measure nodes (R_S and R_M) can be ignored.

This is called a 4-terminal guard measurement.

A 6-terminal guard measurement

Even higher accuracy can be obtained by using two more MUX channels and test nails to make the source voltage and the current sense connections at the scanner. The sense terminal in the voltage source is connected by a separate nail to the source node. The guard node sense terminal of the current meter is connected to the measure node in the same way.

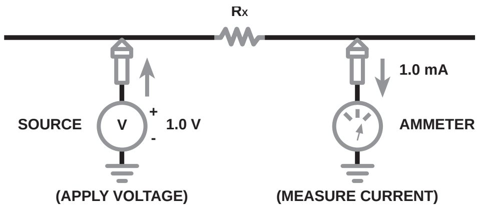
This 6-terminal guard measurement technique is particularly useful when testing low-valued resistors.



Review of 2-, 3-, 4- and 6-terminal measurements

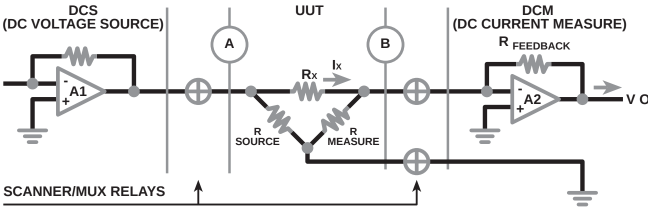
2-Terminal

Ignores the effects of surrounding components.



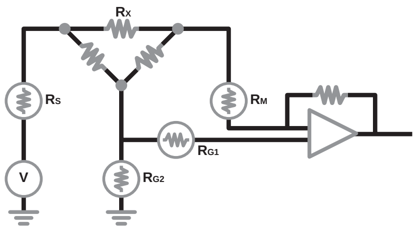
3-Terminal Guard

Takes into account surrounding components, but ignores contact resistance in MUX and scanner.



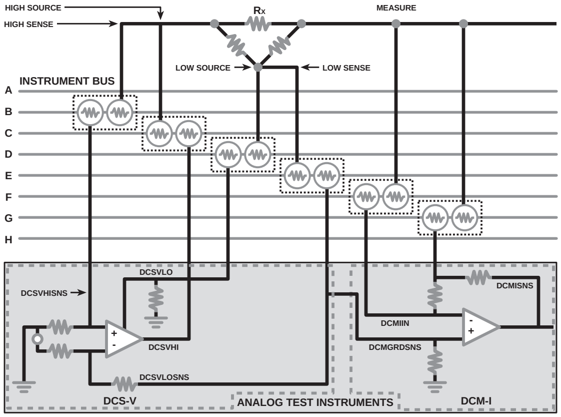
4-Terminal Guard

Takes into account both the surrounding components and contact resistance in MUX and scanner.



6-Terminal Guard

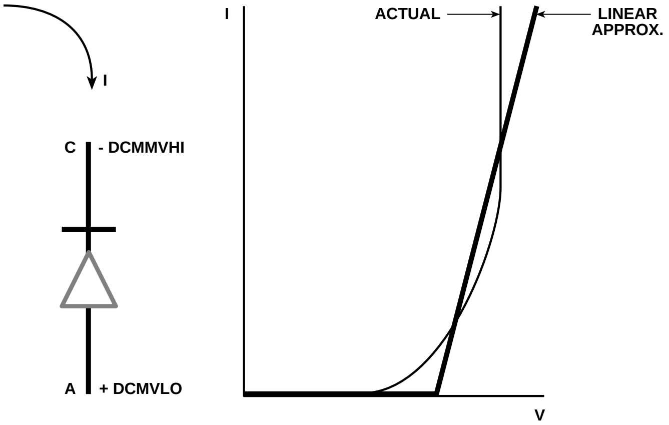
Adds two additional MUX channels to take into account the resistance inherent in the source and measure circuits.



Testing analog devices

Earlier in this chapter, we demonstrated a simple resistor test. The current through the resistor is proportional to the voltage across the resistor, it's a linear relationship. Unfortunately, not all circuit elements possess this type of current-voltage relationship.

Let's examine the relationship between the current through a diode and the voltage across it. As shown in the figure, the relationship is non-linear, and therefore, a different approach must be taken when testing diodes.



Testing diodes

The diode test can be divided into two segments. First, a forward knee voltage test is performed to make sure the diode can operate in the forward bias region. The DC current source injects a negative current (I1) into the cathode and uses the DC voltmeter to measure the voltage across the diode (V1).

The second part of the test calculates the dynamic resistance of the diode and compares it to the expected value. This is done by changing the current setting (I2) and again measuring the voltage across the diode (V2).

The tester approximates the exponential characteristics of the diode operating in its forward bias region with the following equation:

$$\mathbf{Rd = (V2\text{-}V1) / (I2\text{-}I1)}$$

Making complex impedance measurements

Most circuit boards manufactured today contain capacitors and inductors. These components, unlike a resistor which can be tested with DC instruments, need to be tested with AC instrumentation because capacitors and inductors are frequency dependent components. Luckily, Ohm's Law (V=IR) still applies. However, the voltages and currents are now complex values, and the ratio of voltage over current is known as impedance.

Let's examine the test for a 100 nF capacitor.

> The tester writes a program based on the component's value and selects the source amplitude and frequency at which to set the AC voltage supply. The tester also directs the AC current meter to wait a specified time after the voltage has been applied before measuring the current. This time interval allows the capacitor to fully charge before a measurement is made, and will vary depending on the size of the capacitor.

After it measures the current, the tester calculates the impedance across the capacitor and converts these results to a capacitance value using the following formulas:

Impedance = Z = Vsource / Imeasure

Capacitance = 1/(2π * frequency * Z)

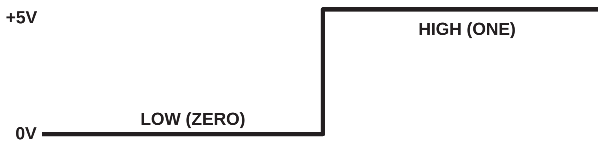
A few other considerations to be taken into account when adjusting test variables for impedance measurements include:

- As the value of the capacitor under test decreases, the test signal's frequency should be increased.
- Increasing the test voltage may turn on neighboring IC chips on the board.

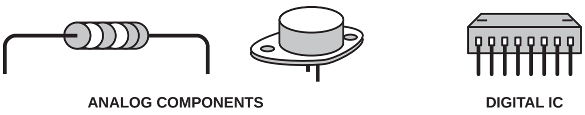
How do digital test instruments access the test nails?

First, consider some of the typical features of digital components.

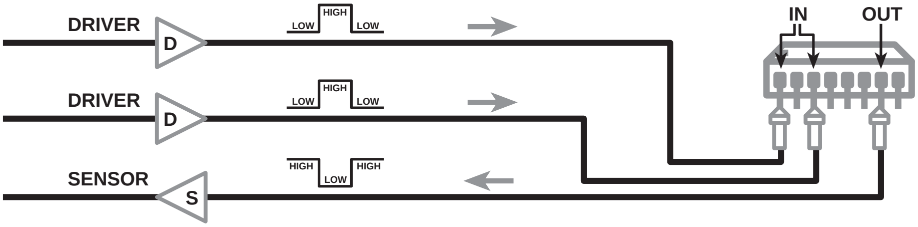
- Digital components deal with only two voltage levels: a LOW (ZERO) level and a HIGH (ONE) level, as shown in the example below. The test program can define these two logic levels before digital testing begins and then simply refer to the two levels as HIGHs and LOWs throughout the digital test program.



- Digital ICs (Integrated Circuits) typically have many more pins than analog components such as resistors, capacitors, transistors, etc. It would be impossible for the tester to keep switching analog source and measurement instruments from one IC pin to another during a test.



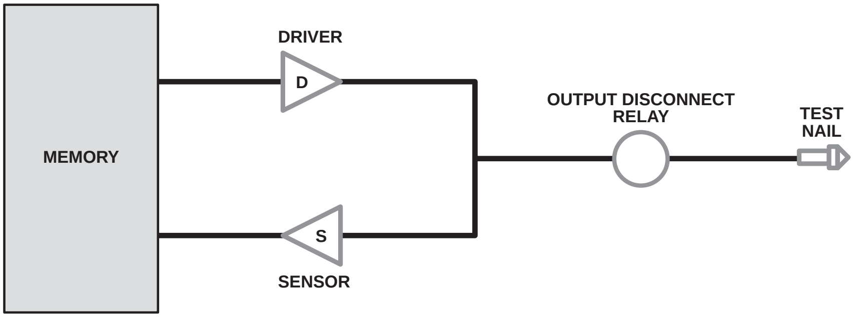
So, the tester has a set of digital drivers that it uses to drive the IC inputs to desired states and a set of digital sensors to verify the logic levels at the IC outputs.



Digital driver/sensors

The drivers and sensors (D/S, for short) exist in pairs. The output of a driver is always tied to the input of a sensor. The driver and the sensor, however, are separately controllable by the program.

Digital inputs are usually connected to other digital outputs. Therefore, a node may have to be driven to a specified state during one test step, and that same node may have to be sensed to determine its output state during another test step.



The Driver/Sensor pair makes it easy to do this. To drive a node to a particular state, the driver portion of the D/S pair is enabled and its output is forced to the specified state. Similarly, to check a logic output at that same node, the driver is disconnected (electrically) from the node and the sensor portion of the D/S pair is enabled.

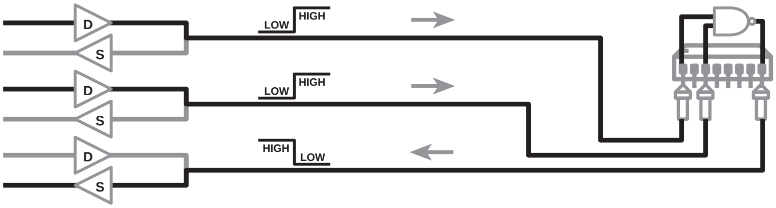
Whenever the driver is enabled, the sensor portion can be either enabled to sense the driver output or it can be disabled.

On the next page, we'll look at some simple test language statements that are used to control the driver/sensors.

Programming the driver/sensors

To illustrate how the program controls the driver/sensors, let's look at a simple example. We'll use the 2-input NAND gate shown below.

The output of that NAND gate is LOW only when the two inputs are HIGH. Any other combinations of inputs causes the NAND gate output to be HIGH.



Before any digital testing begins, the test program:

- Applies power and ground to the PCB under test.
- Defines the HIGH and LOW logic levels.
- Tells the system which nails will be used during the test.
- Tells the system how fast and at what time to apply the logic inputs and to sense the logic outputs.

For the single NAND gate example we've chosen, the following test program statements will check the operation of the gate for all four combinations of inputs.

	Drive		Sense
Program statement	Input Nodes		Output Node
	(A)	(B)	(C)
IC(A,B) IH(A,B) OS(C) OL(C);	H	H	L
IL(B) OH(C);	H	L	H
IL(A) IH(B);	L	H	H
IL(B);	L	L	H

NOTE: Each row in the table represents a test vector.

The commands IC, IH, and IL connect the drivers to the specified nodes A and B and assign values to them. The OS, OH, and OL commands connect the sensor to node C and specifies what logic value to expect. All the driver and sensor values remain unchanged until specifically changed by a program statement.

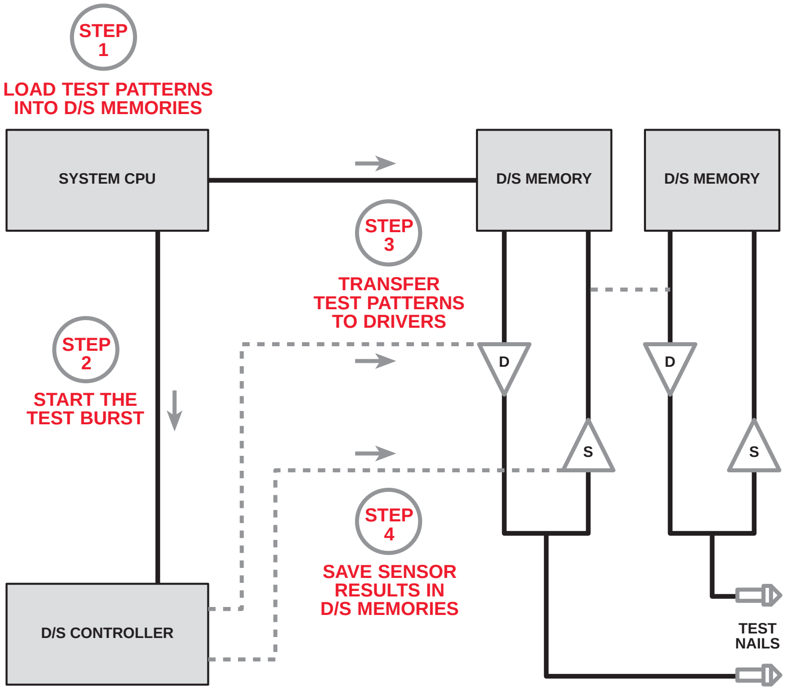
The driver/sensor system

The tester's Central Processing Unit (CPU) loads a series of test patterns from the test program into a bank of memory cells attached to the driver/sensors (D/S). Each D/S has its own memory bank.

To start a test, the system CPU enables a special controller, which handles the transferring of these test patterns to the drivers and the storing of sensor results into the D/S memories.

This high-speed digital test procedure, illustrated below, is called a test BURST.

If a device fails or if specified by the user, the CPU transfers the results from the D/S memories back to the main memory for analysis, after the burst is complete.


STEP 1
LOAD TEST PATTERNS INTO D/S MEMORIES
SYSTEM CPU
D/S MEMORY
D/S MEMORY
STEP 3
TRANSFER TEST PATTERNS TO DRIVERS
STEP 2
START THE TEST BURST
D
D
S
S
STEP 4
SAVE SENSOR RESULTS IN D/S MEMORIES
D/S CONTROLLER
TEST NAILS

Nail types

In addition to the standard test nails, the driver/sensor system contains several other nail types, which perform various digital testing functions:

- **Fast nails -** These are the test nails that connect to the driver/sensors, (or to the analog instrument bus) and can execute up to five million test steps per second.
- **Clock drive nails -** These are special nails that you can use to supply programmable clock inputs to the UUT. You can program drive and sense strobes to occur at any time increment during the clock cycle.
- **Clock sync nails -** Clock sync nails are used to handle test procedures that require each digital test step to be synchronized with a clock on the UUT. This could occur, for example, if the UUT clock cannot be disabled. The selected sync signal controls the internal timing of the tester during the burst.
- **Trigger nails -** Trigger nails can be used to synchronize a test with a particular bus cycle or other unique events occurring on the UUT. The tester monitors control pins on the UUT and aligns its test strobes to coincide with specific conditions.

All of these nail types are present in the system, however only the fast nails are connected to the driver/sensors. The clock and trigger nails do not support normal testing. Therefore, an additional fast nail may have to be connected to a clock/trigger node for test purposes.

Let's examine how these nails can be used to perform especially high-speed (FAST) testing.

Testing fast devices

The fast, clock, and trigger nails described on the previous page are part of a high-speed subsystem that is especially suited to testing VLSI (Very Large-Scale Integration) components. This subsystem can perform VLSI testing much easier than the general-purpose digital test hardware.

When a FAST statement occurs within a burst, the standard controller passes program control to a high-speed controller. During the FAST section of the burst, the high-speed controller executes the FAST test steps independently of the standard controller, usually at a much higher test rate.

Tests run in FAST mode use the high-speed subsystem to furnish the test step timing, clock signals, and test synchronization.

Clocking

Many VLSI devices have a clock pin that must be driven with a pattern of alternating highs and lows. In the high-speed subsystem, a clock generator performs this clocking function.

Triggering

The high-speed subsystem's triggering function allows you to specify that one or more nails must reach a particular state before the test step can execute. If more than one trigger nail is specified, the test step waits until all the trigger conditions are met.

Clock Synchronization

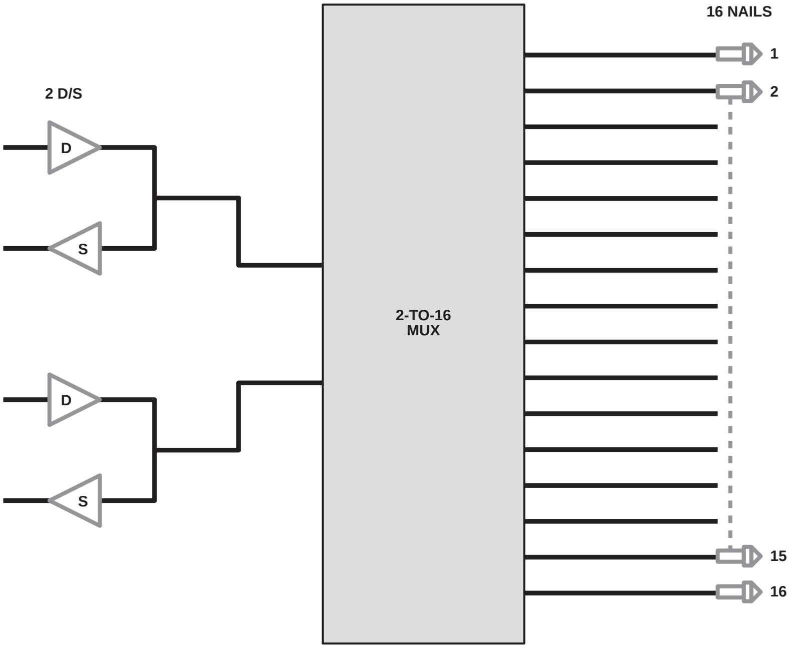
If necessary, the high-speed clock module can be synchronized to a signal on the UUT. For example, if a UUT clock cannot be successfully disabled, the tester may simply synchronize the test program to the on-board clock.

One more thing about the driver/sensors

Testing large PCBs that have many digital ICs requires a test fixture with many digital test nails. However, not all of these nails are used at the same time. (All of them contact the board at the same time, of course, but the tester uses only a few of them during each test.)

In fact, during each test (or burst), the system uses only those nails that can affect the IC being tested.

So, for economical reasons, each test nail does not usually have its own dedicated driver/sensor, but share a few driver/sensors with a group of other nails. This technique is called driver/sensor multiplexing. Chapter 4 explains in more detail the concept of driver/sensor multiplexing.



How does the tester isolate and test digital components?

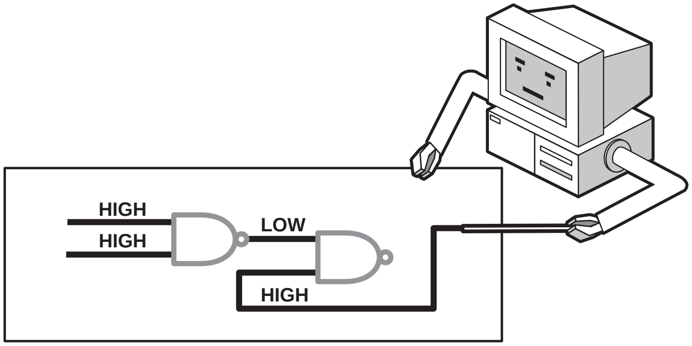
Earlier in this chapter, we saw how a simple 2-input NAND gate could be tested:

- First, power (and ground) had to be applied to the IC to make the circuitry within the chip operational.
- Then, all possible combinations of logic inputs (highs and lows) were applied to the IC while at the same time the output was checked for the proper logic level.

Now, what happens when this IC is connected in a circuit?

For one thing, since power must be applied to the IC, circuit interconnections on the board will cause that power (and ground) to be applied to the rest of the ICs on the board, as well. Remember, we did not apply power to the board when testing analog components.

Therefore, when the tester tries to apply a logic level to the IC input, it may find that the IC input pin is being held in the opposite state by the output of another IC. For example, the test program may call for the tester to apply a high input to a node at the same time that node is held low by another IC output.

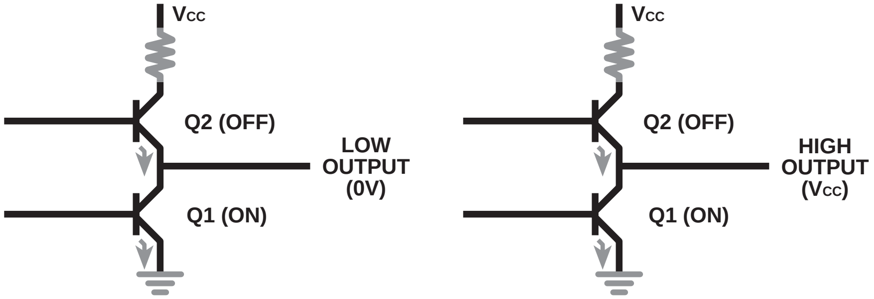


The digital drivers in the tester handle this problem by momentarily forcing the IC input to the desired state, regardless of which state it is being held in by another IC.

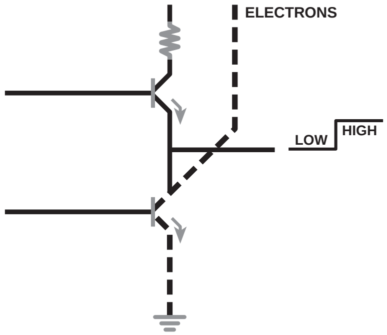
This technique of momentarily overriding an IC output is called backdriving.

What is involved in backdriving?

Consider the typical output stage of an IC, such as the TTL gate shown below. Note: TTL (Transistor-Transistor-Logic) refers to the type of circuit architecture used in the design of the gate.



A low output occurs when transistor Q1 is conducting (is on) and Q2 is off. To override this condition temporarily and develop a high output (say, about 3.5 V), the tester forces a current pulse of typically 200 mA (but could be up to 500 mA depending on the device technology) back through Q1. This current flowing through the emitter-to-collector resistance of the transistor develops the high output.



Similarly, a high output exists when Q2 is conducting and Q1 is off. If the tester wants to force the output low, it applies a low level to the output and sinks the resulting current flowing through Q2.

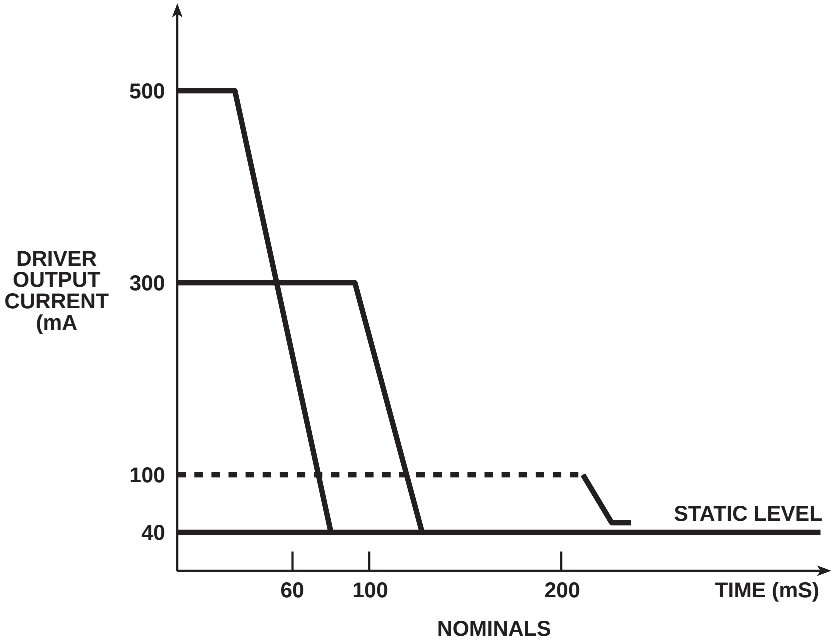
Remember, the purpose of backdriving is to force a desired logic input to the IC-under-test, regardless of what any other IC outputs might be doing.

Doesn't this backdriving current harm the IC?

Not if the current is applied for only a very short time.

Since digital tests are conducted at high speeds, the test current pulse is normally much less than 10 ms wide (typically 5-10 μs). This brief current pulse does not damage the IC.

If for any reason, however, a driver remains on too long for the particular amount of current that it's driving, a sense circuit in the driver automatically reduces the current to around 40 mA, to protect both the IC and the tester electronics.



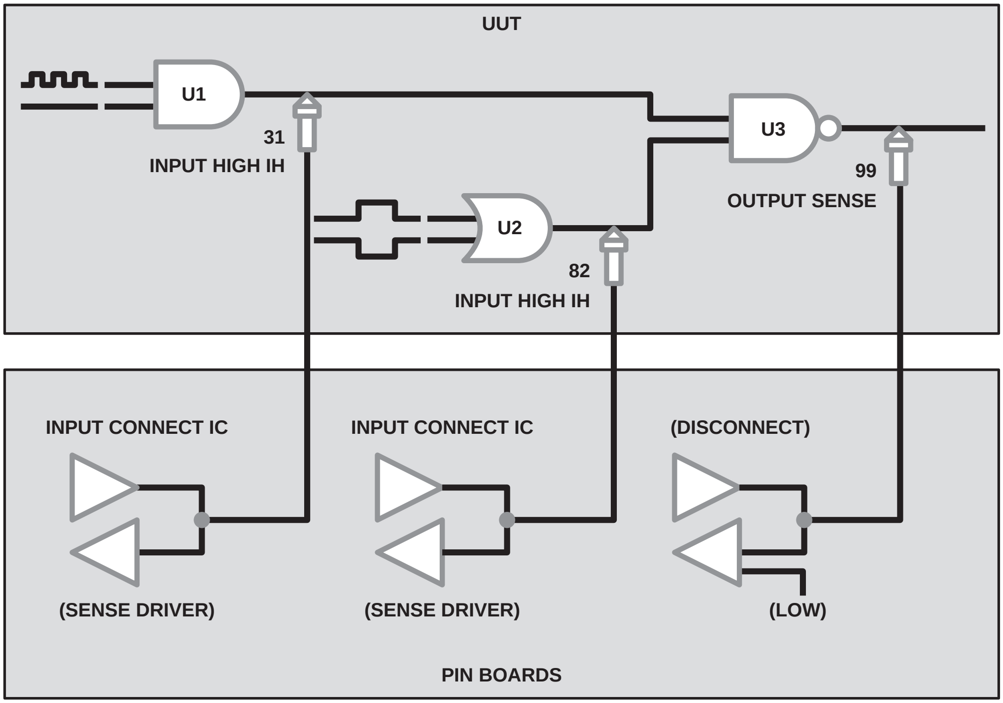
Now, back to the in-circuit digital test example.

As we've already seen, before any digital testing begins, the test program:

- Applies power and ground to the board.
- Defines the high and low logic levels.
- Specifies the timing for the test inputs and outputs.
- Tells the system which nails will be used during the test.

After this is done, test bursts can be executed to test each IC on the board. Test vectors consisting of known good inputs drive the component to an expected state. The tester verifies that the component has been driven to the expected state by sensing the output.

The following figure shows the simple NAND gate we've seen before, as it would typically appear in a circuit. The digital drivers in the tester will momentarily force the inputs to NAND gate U3 to the desired state regardless of the output states of U1 and U2.

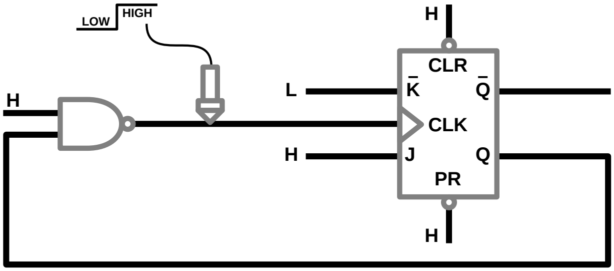


So that's all there is to digital testing?

Not quite.

Backdriving is certainly the key to digital testing but as you might expect, other problems can arise. For instance, consider what can happen when testing an IC that's part of a feedback loop.

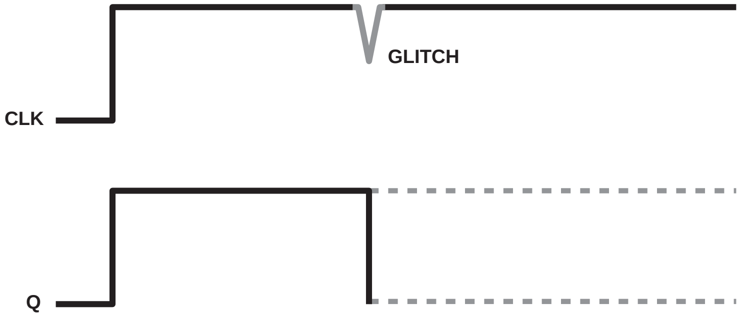
Let's say that the flip-flop shown below is initially cleared and is set-up to toggle to the opposite state whenever a high-going pulse is applied to its clock (CLK) input.



Part of the test on the flip-flop would be to pulse the CLK input and make sure that the flip-flop output changes state.

However, when this happens, the following unwanted action also takes place:

- The sudden change in the flip-flop output (high-to-low) immediately feeds back to the CLK line, driving it low.
- After a short time, the driver connected to the CLK line recovers and drives that node back high (where it's supposed to be).

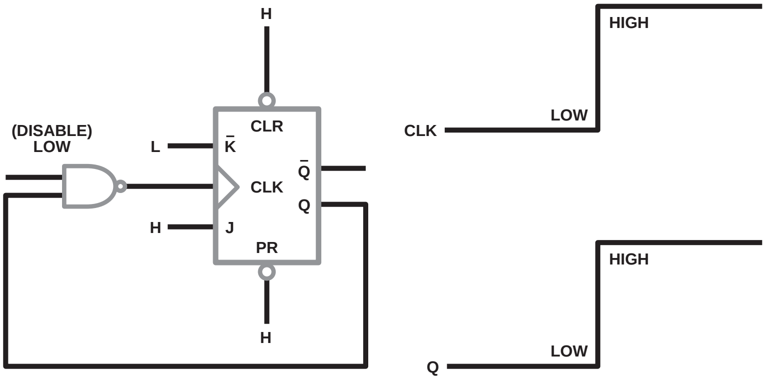


So, what's the problem?

The problem is the momentary glitch that appears on the CLK line while the driver is recovering. That glitch (depending on how big it is) might toggle the flip-flop back to the state it was in before the test.

If this happens, the tester will assume that the flip-flop did not toggle and, therefore, failed the test.

To prevent these glitches from becoming a problem, the tester analyzes the surrounding circuitry and tries to inhibit every pin that is being backdriven, for example, any device pin in the feedback loop which may interfere with the test.



To protect against unwanted glitches during a test, GenRad's in-circuit testers disable all 3-state devices by placing them in their high-impedance state and inhibit all other devices (except the IC-under-test) by forcing their input pins to a state that effectively inhibits their operation. A 3-state output can be high, low or in a high-impedance state.

Backdriving and inhibiting digital devices

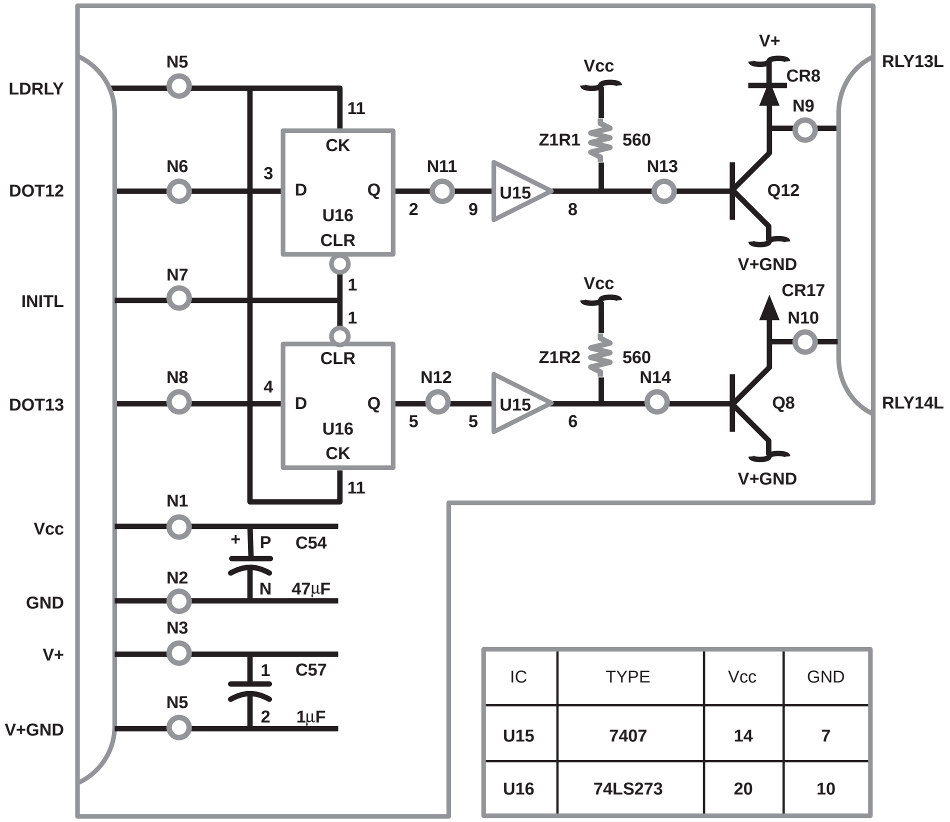
Here is an example of how digital devices are backdriven and inhibited.

In the circuit shown below, you can test U16 as if it were the only device on the board. U15, however, presents a potential testing problem because of its connections to U16. You cannot disconnect the flip-flop outputs to U15. They continue to drive the inputs in parallel with the tester during the test.

Therefore, if one of these outputs happens to be high when the tester needs to supply a low to a U15 input, it must override the flip flop by supplying enough current to pull it briefly to the desired state. This isolation technique is called backdriving.

The flip flop's clock input presents an additional problem. Digital noise from the test of U15 could cause the flip flop to change state during the test. This would briefly give a false output from U15 despite the tester's attempt to backdrive the flip flop output.

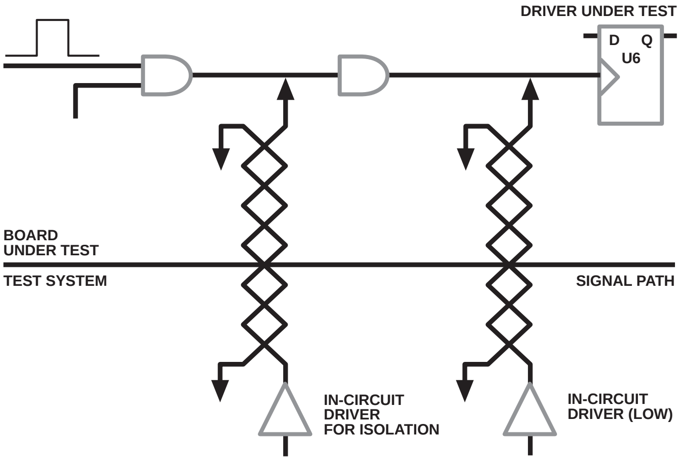
Therefore, the test invokes another isolation technique called inhibiting to hold the flip flop in a known state. From the system IC library, the tester finds that it can hold the flip flop output stable by pulling the CLR line low. Since CLR overrides all other inputs, the outputs cannot change state and cannot disturb the testing of U15.



IC	TYPE	Vcc	GND
U15	7407	14	7
U16	74LS273	20	10

Multiple Level Digital Isolation

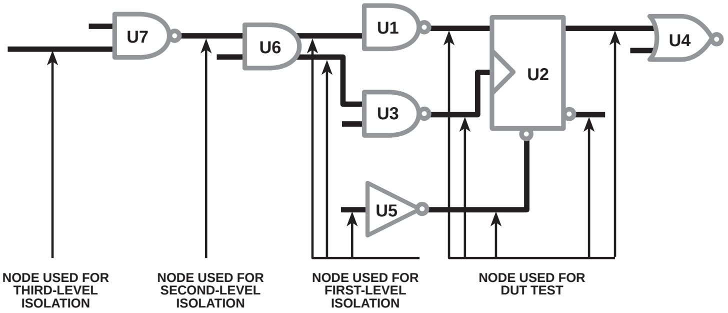
To isolate ICs, the tester can apply signals upstream from the device-under-test (DUT). In the example on the previous page, we saw how the flip flops (U16) could be isolated from the drivers (U15) by holding the CLR line of U16 low. As shown in the following figure, adding a second driver to hold the input of the device driving the DUT keeps its output from changing states, and blocks unwanted signal changes from getting to the DUT's input.



The Multiple Level Digital Isolation (MLDI) software chooses device isolations separately for each test. This means that the isolation chosen for a device depends only on the conditions for that particular test. It is independent of the isolation chosen for that same device in a previous test.

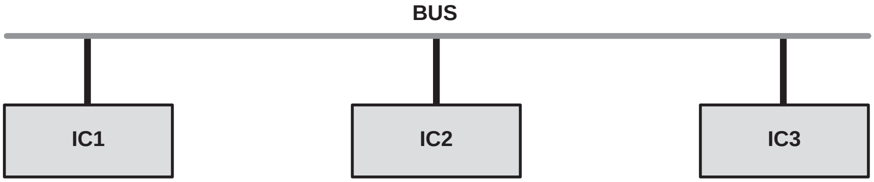
MLDI lets you specify the number of isolation levels from the DUT. The following figure shows the concept of MLDI. U2 is the DUT. The tester connects nails to its inputs and outputs. The input of U1, U3, and U5 need to be isolated from the DUT.

- First-level isolation adds the connections to the inputs of U1, U3, and U5. The input nodes on U1 and U3 can still be driven by U6. The U5 input node has no device outputs driving it, so no further isolation is needed.
- Second-level isolation adds a connection to the input of U6, but U7 can still drive the U6 input.
- Third-level isolation adds a connection to the input of U7. No outputs drive U7, so no further levels of isolation are needed.

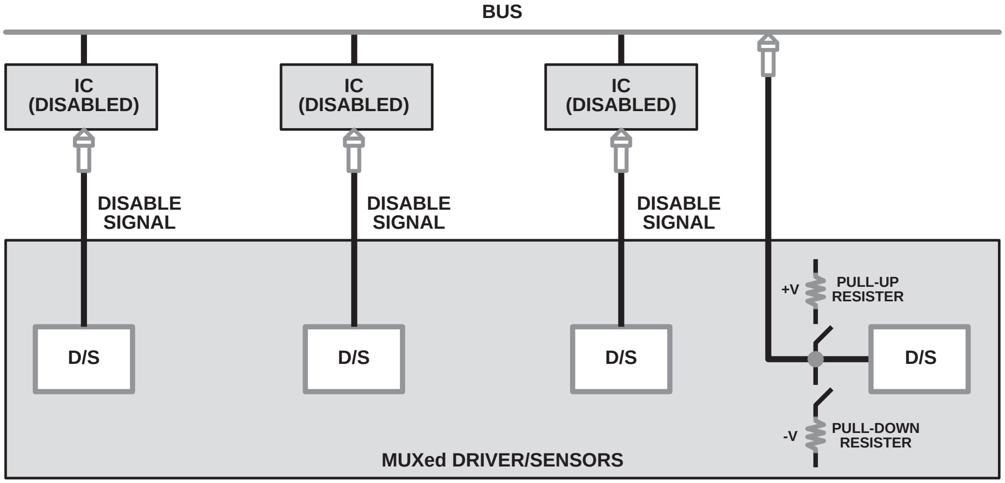


Any other problems?

Another special situation arises when testing bused devices, i.e., several devices all tied to a common bus.



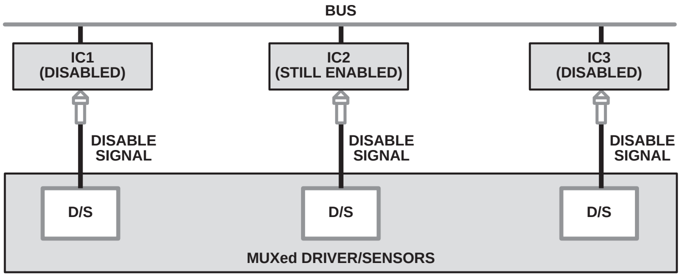
Before testing each individual device on the bus, the tester first disables them all, then connects the bus to a pull-up and then a pull-down resistor to make sure that the bus is not stuck in one state (either high or low).



If this pull-up/pull-down test is successful, the tester then checks each device individually to see if each one can control the state of the bus.

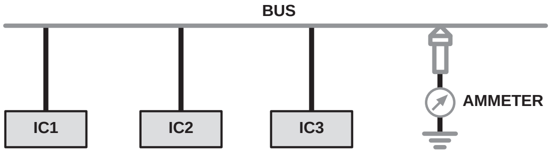
What happens if a bus is stuck in one state?

Good question. How can you tell which innocent-looking device refuses to be disabled? The device outputs are all tied together, so any one of them could be forcing the bus to this error state.



GenRad uses something called a BUSBUST™ technique to find the bad bus device. The technique follows these steps:

1. With all devices still "disabled", the tester measures the current at the failing bus node.
2. Then, it turns on one device at a time and applies logic inputs to that device to try to drive the bus node in the direction of the failure. For example, if the node is stuck low, the logic inputs to the device will cause its output to go low.

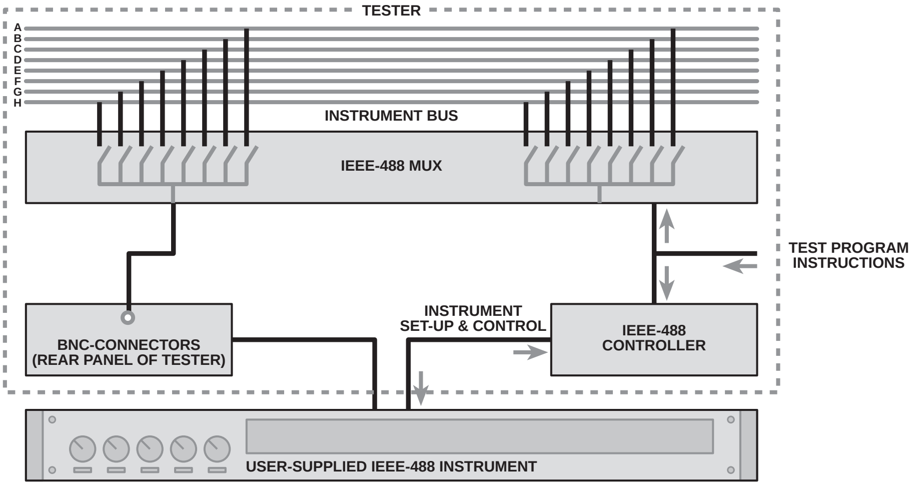


3. The tester again measures the current at the node.
4. If the current changes significantly, the device is deemed OK. However, if the current remains essentially the same as when the device was "disabled", then the IC is assumed to be bad.

Can I use external test instruments?

Yes, provided that these instruments conform to industry specifications such as IEEE-488, GPIB, VXI, and RS-232. These specifications define protocols and interface requirements used to communicate and program the devices.

All of GenRad's in-circuit testers have an optional IEEE controller and multiplexer. Under test program control, the MUX can connect external IEEE-488 devices to the same instrument bus as the standard analog instruments, and the controller can handle the initialization and operation of these instruments.


TESTER
A
B
C
D
E
F
G
H
INSTRUMENT BUS
IEEE-488 MUX
TEST PROGRAM INSTRUCTIONS
INSTRUMENT SET-UP & CONTROL
BNC-CONNECTORS (REAR PANEL OF TESTER)
IEEE-488 CONTROLLER
USER-SUPPLIED IEEE-488 INSTRUMENT

In the next two chapters, we'll take a closer look at an actual in-circuit test system: its hardware, software, and user interface. In Chapter 6, we'll investigate how you can put the in-circuit tester to work for you. As board densities increase and components become more complex, using in-circuit testers requires some strategic planning prior to production testing of boards.

Chapter Four

A Look at an In-Circuit Tester

Chapter Four

A Look at an In-Circuit Tester

Meet a real-live in-circuit tester

Now that you're familiar with some of the basic in-circuit test techniques, let's see what a real tester looks like. Shown below is GenRad's tester, which contains all the major components you were introduced to in Chapter 3, along with a few other components that you'll learn about in this chapter.

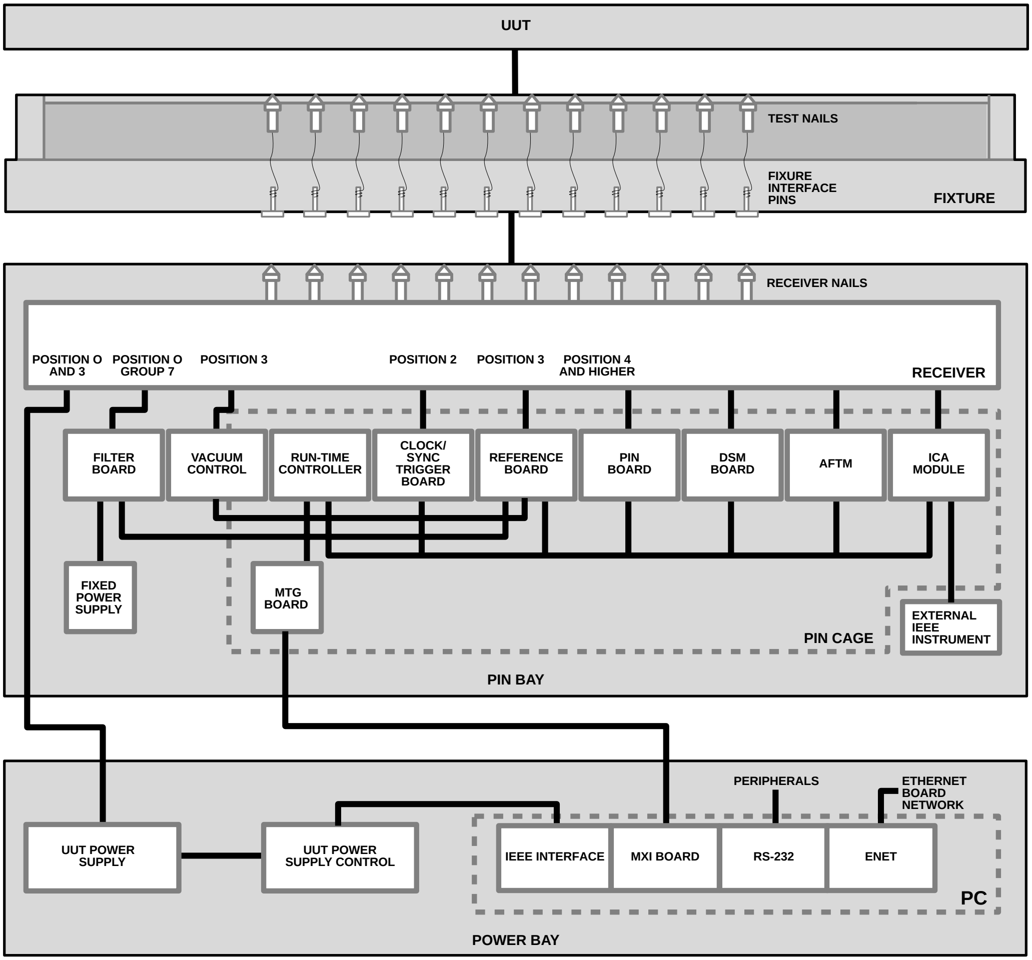
We'll take a closer look at each of these components - what they consist of, what they do. And, we'll do it in a logical order by following the test path: from the UUT*, through the fixture and receiver, to the pin boards, the test instruments (both analog and digital), to the computer (PC) that controls the whole operation.

---

*Throughout the rest of this book, the board being tested will be referred to as the Unit-Under-Test, or UUT.

The big picture

To get you oriented in this logical test path, here's a top-level view of how the tester's major components fit together.



- The UUT is pulled down onto the bed-of-nails by a vacuum or a pneumatic fixture. This allows the fixture nails to come in contact with the circuit nodes on the UUT.
- The fixture probes on the fixture are wired to receiver pins that attach to a series of connectors on the receiver.
- The other end of the receiver connectors plug into pin boards, which are used to route signals to and from analog and digital test instruments via various buses along the tester's backplane.
- The UUT power supplies connect to a heavy-duty set of receiver pins to power-up the UUT for digital testing and analog functional testing.
- The PC controls the entire test operation through its software program called the Run Time System.

Now, let's look at these components in more detail.

The test fixture

The fixture contains the probes that come in contact with all the circuit nodes on the UUT. As we follow the test path, we'll see that the other end of these fixture nails eventually connect to the test instruments, through a series of software-controlled relays, or switches (called scanners and multiplexers).

Prior to starting a test, the following actions occur:

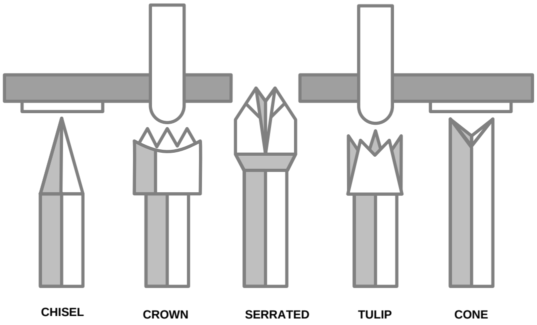
- The operator places the UUT on the fixture. Alignment pins on the fixture ensure that each circuit node is precisely aligned over a test nail.
- Under software control, the tester applies a vacuum to the fixture, which pulls the UUT down onto the spring-loaded test nails. The test nails are now in contact with the UUT nodes.
- The bottom of the fixture nails are wire-wrapped to pins on the fixture interface board. When the fixture is pulled down, these fixture pins mate with corresponding pins on the receiver.

The probes in the fixture that contact the UUT are called fixture nails. The nails on the system side that connect the test instruments to the UUT are called receiver pins.

More about fixtures

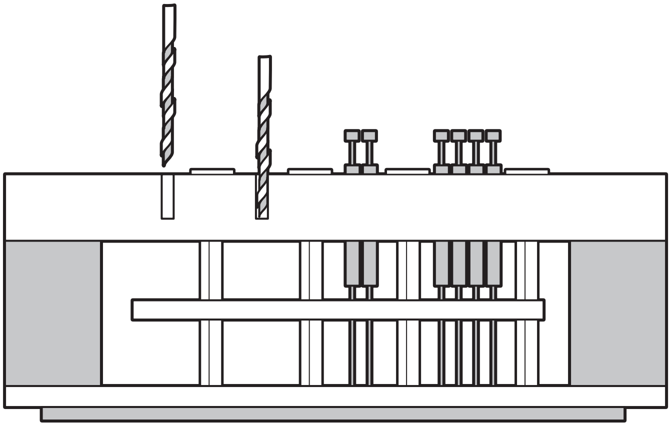
Much could be said about the different types and complexities of fixtures, as well as the different types of test nails available. But, that would be beyond the scope of this book. Instead, we'll just briefly describe a few of the available features.

- **Types of fixture probes -** To ensure proper contact with the circuit nodes, test nails can have various types of styles. They can be cone-shaped, cupped, crowned, serrated, etc. The type of nail head used depends on the condition and the type of the node. For example, pointed tips can penetrate oxides and contaminants. Smooth tips will avoid indentation. Some tips are made to contact plated-through holes; others are specially designed to contact wire-wrap pins or component leads.



- **Dual-well fixtures -** Dual-well fixtures have two separate bed-of-nails (wells) which are controlled independently from one another. While one board is being tested on one well, another board can be loaded on the other well in preparation for its test. Consequently, dual-well fixtures can be used to speed-up testing in a high-volume test environment.

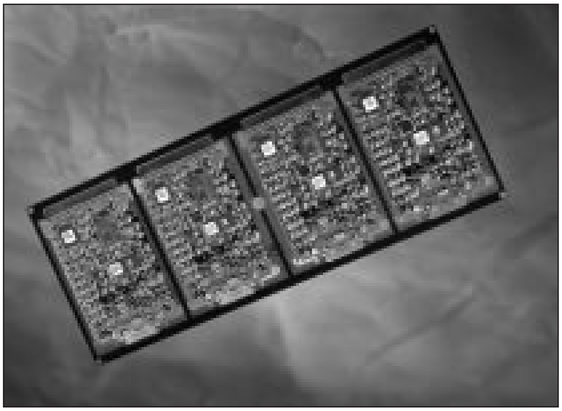
- **Wireless fixtures** - Wireless fixtures eliminate wiring by using a PCB to connect receiver pins to a fixture probe. A wireless fixture can:
  - Improve reliability and signal integrity.
  - Reduce weight and size.
  - Be less expensive for multiple fixtures of the same type.



WIRELESS FIXTURE

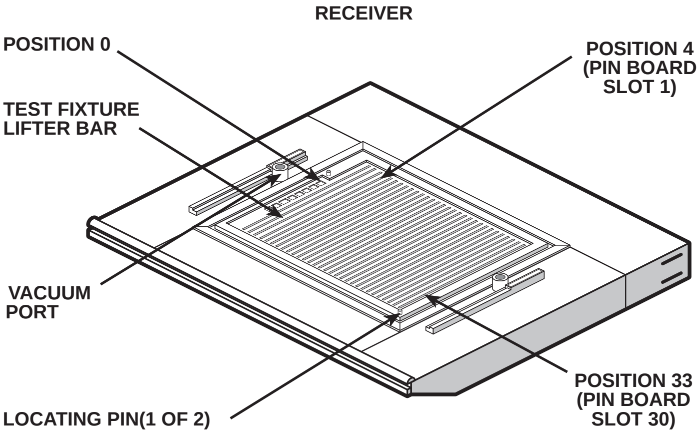
Other fixture options

- **Converters** - Converters are used to adapt an existing test fixture to a different mechanical receiver interface.
- **Panel-Test™** - Panel-testing can save both time and money when testing large quantities of smaller boards. In panel-testing, multiple boards are manufactured on a single panel and placed on a test fixture designed for that panel. Once the operator initiates the test, the multiple boards are tested sequentially without any further operator intervention. The system software reports the test results for each board separately.



The receiver

As explained earlier, the fixture probes are wire-wrapped to receiver pins on the fixture interface board. These pins then connect to corresponding pins on the receiver.



The receiver interface contains the following major items:

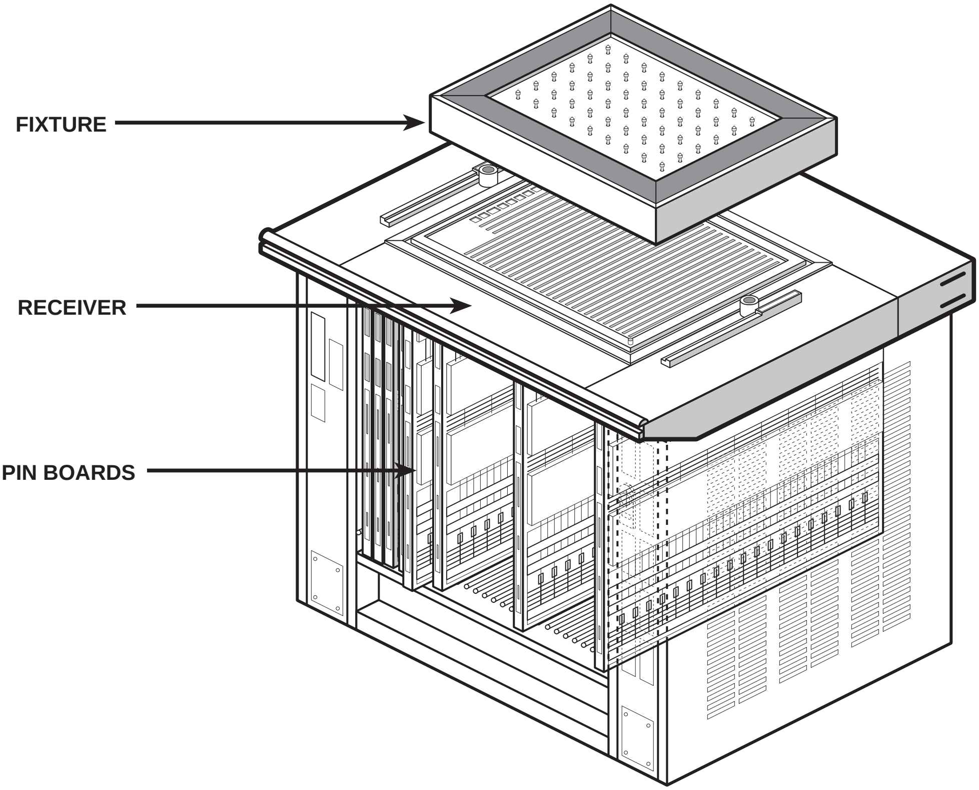
- Vacuum port(s) to pull the UUT down on to the fixture.
- Heavy-duty connector pins in position 0 that are used for connecting the tester's UUT power supplies to the fixture.
- Connectors used to interface the pin boards (which connect to the tester electronics) to the UUT. (More about UUT power supplies and tester electronics later.)
- Locating pins that fit into locating holes in the fixture to ensure that the fixture and receiver are properly aligned.
- A fixture lifter to make or break the mechanical connection between the fixture and the tester receiver.
- A receiver gasket to prevent vacuum leaks.

The pin boards

We've now gone through the mechanical interface between the UUT and the tester. Now, it's time to look inside the tester at the tester electronics.

What's behind the receiver? The answer is the pin boards, which form the main interface between the receiver and the tester electronics.

The pin boards plug into a pin cage which is located in the pin bay. The pin boards also plug into the receiver connectors which are located on top of the pin bay.

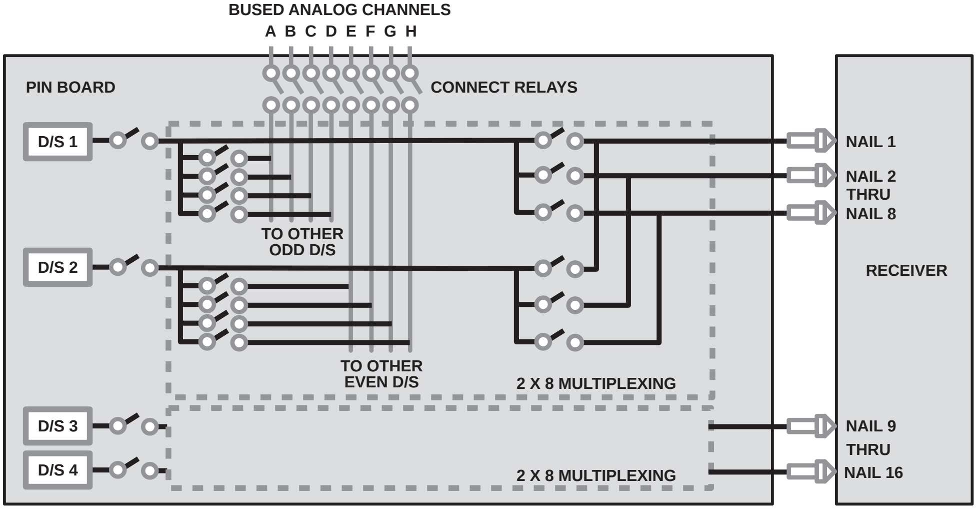


The pin boards perform several important functions:

- They contain the driver/sensors that are used for digital testing.
- They can connect either the analog test instruments or the digital test instruments (driver/sensors) to the test nails. The pin boards perform the analog scanning function described in Chapter 3, where selected analog instruments are connected to specified test nails
- They also perform a pin-multiplexing function, which allows the system's analog and digital test instruments to be shared by all the test nails.

A more detailed diagram of the pin board

The following figure shows a more detailed diagram of the pin board and illustrates how it performs its various functions.



Each pin board contains 16, 32, or 64 digital driver/sensors. These driver/sensors can drive or sense any pre-programmed logic level.

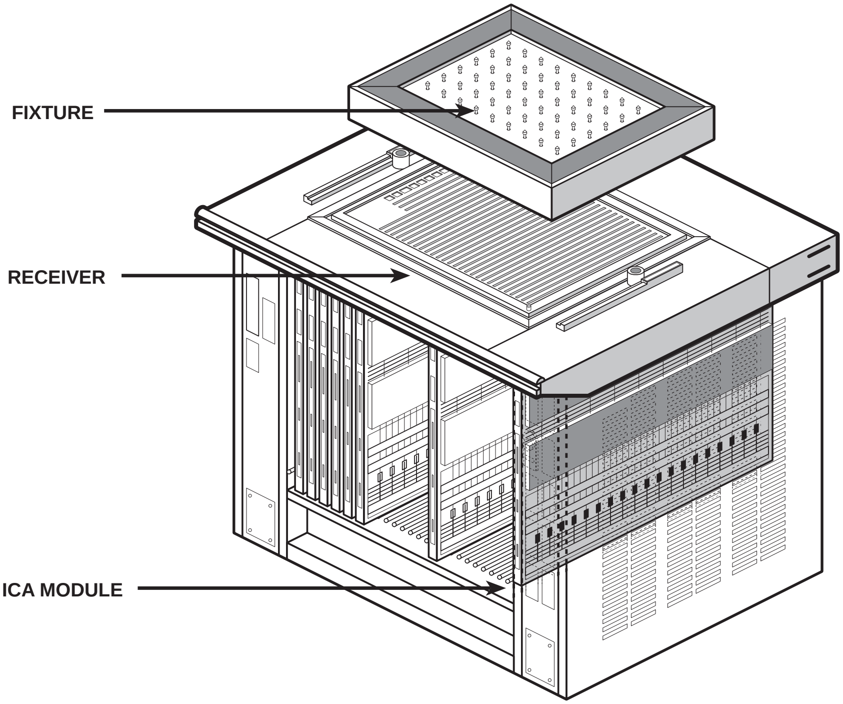
The 8 analog channels (or busses) A through H are available to each pin board. A set of relays on the pin board direct either the analog bus or the driver/sensors to the pin multiplexer, but not both at the same time. Under software control, any analog test instrument can be connected to any of the 8 analog bus channels.

The pin multiplexer is a set of programmable switches that multiplexes the selected analog or digital test instrument to a large number of test nails. By means of the pin-multiplexing technique, two driver/sensors are shared by every 8 or 16 test nails. That is, in each 8-nail or 16-nail group, only two test nails can be used at the same time for UUT testing. We will learn more about this procedure later. Nail assignment software in the tester ensures that test nails are assigned to nodes in such a way as to avoid multiplexing conflicts.

In-Circuit Analog (ICA) module

The ICA module contains the analog source and measure instruments, and two instrument multiplexers. One multiplexer is used for the standard test instruments contained on the ICA module. The other multiplexer can be used to connect external IEEE-488 test instruments to the tester.

The ICA module consists of one board (with optional plug-in board) and is located in the pin cage, as shown below. The ICA contains the following source and measure instruments:



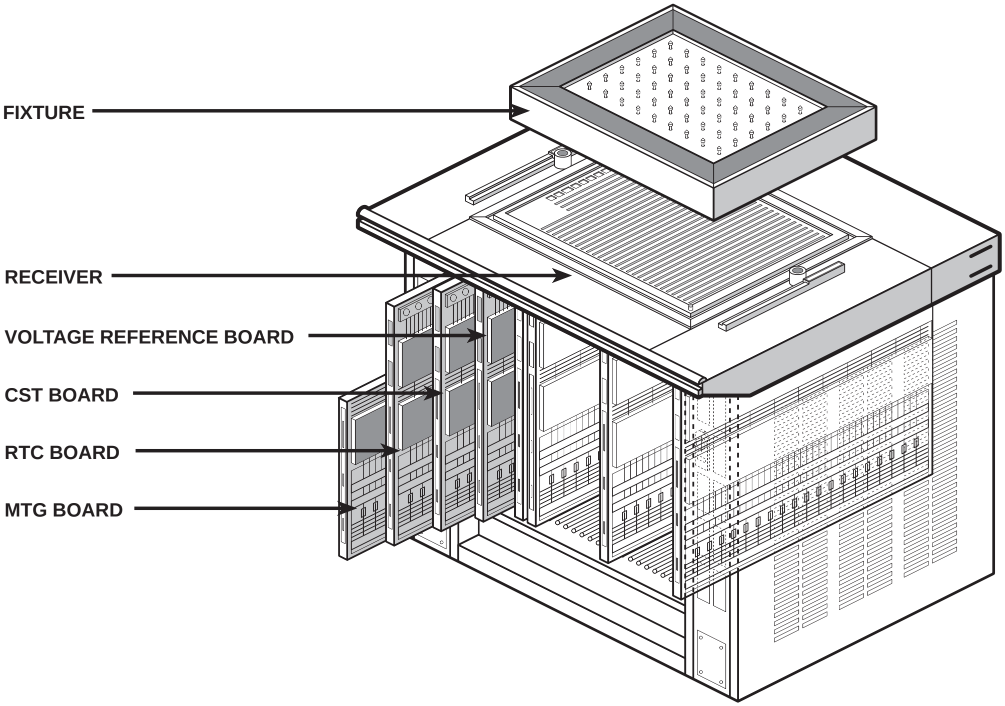
- Source 1 - AC or DC voltage or current source
- Source 2 (High voltage option) - Voltage or current
- Voltmeter 1 - AC or DC
- Voltmeter 2 - DC
- Ammeter - AC or DC
- Arbitrary Waveform Generator (AWG) - AWG can create standard sine, square, triangle, or user-defined waveforms.
- Digital Multimeter (DMM) - The DMM can be used to make a sequence of voltage or current measurements and store them in an array.

The ICA's instruments can be used both as stand-alone analog resources but also can be synchronized with digital resources for mixed signal testing:

- Stand-alone instruments, for example, can be used to measure shorts, opens, RLC components, and transistor parameters.
- As a mixed-signal resource, the ICA is synchronized to the tester's digital subsystem. This allows the ICA analog instruments to be triggered and monitored from within a digital test burst.

Other boards in the pin bay

In addition to the Pin Boards and the ICA module, here are some of the other boards contained in the pin cage:



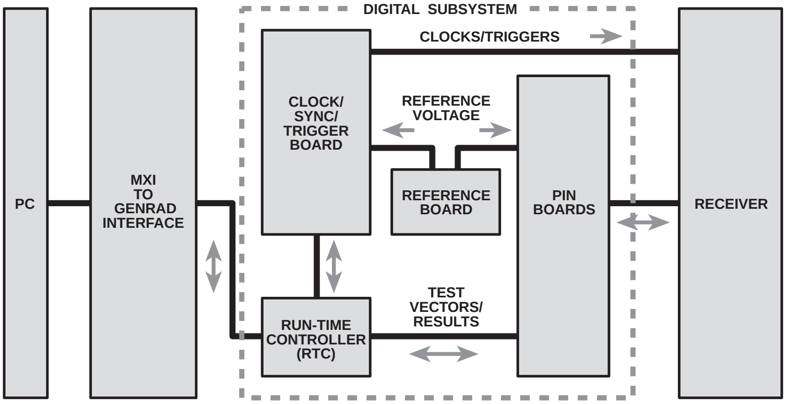
- The MTG (MXI To GenRad) board is used to interface the tester's CPU (Central Processing Unit) to an internal bus that connects to all subsystems in the tester.
- The RTC (Run Time Controller) board controls the tester's digital subsystem (timing, burst control information, and program flow).
- The CST (Clock/Sync/Trigger) board generates the clocking and timing signals needed for complex digital testing.
- The Voltage Reference board supplies programmed DC reference voltages (logic levels) for the digital driver/sensors.

Other optional boards may be used for special applications. For example:

- The AFTM (Analog Functional Test Module) supports a wide variety of analog functional test applications, such as testing D/A converters and measuring frequencies and pulse widths.
- The DSM (Deep Serial Memory) board supplies extra memory for test applications that require many digital test vectors.
- The CFB (Custom Function Board) can be custom-designed to provide specific hardware for a user's special test requirements.

Digital subsystem

Beyond the driver/sensors is a whole digital subsystem that controls each digital test burst. You've seen some of the boards that comprise the digital subsystem when we looked at the boards contained in the pin cage. Shown below is a simplified block diagram of how those boards interact to form the digital subsystem.



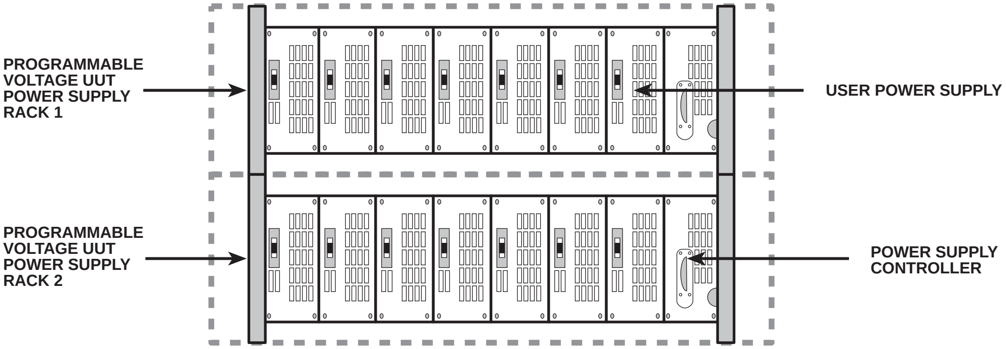
The digital subsystem works something like this:

- Prior to a test burst, the Run-Time Controller (RTC) receives commands and data from the PC through the MTG board.
- The RTC then processes this data and transfers burst control information to local memories on the pin boards. It also transfers timing information to the Clock/Sync/Trigger (CST) board. As the test progresses, the RTC controls program flow throughout the digital subsystem.
- The voltage reference board supplies the reference logic levels used by the driver/sensors and by the timing circuits on the CST board during the test.
- The Clock/Sync/Trigger board provides all the clocking and timing signals needed by the driver/sensors to test the UUT. During each test step in a burst, a precisely timed driver strobe signal causes the drivers to assume the states specified by the test vectors. After a specified interval, a sensor strobe causes the sensor outputs to be sampled.
- The pin boards contain the driver/sensors, each with local memory. At each test step, the test vector data stored in local memory determines whether a driver/sensor (D/S) drives the node, senses the node, or both. When the D/S drives a node, data in local memory specifies whether it will apply a logic high or a logic low. When the D/S senses a node, other data specifies the expected logic state. Still other stored data can disable a driver, disable a sensor, or cause the driver/sensor to repeat the previous function.

UUT power supplies

The UUT must be powered up during digital testing. Power must also be applied to the UUT when functionally testing analog circuits. For example, to test an operational amplifier requires that the op amp be powered up. The tester contains programmable voltage power supplies and some fixed power supplies for this purpose.

All programmable power supplies and their controllers are located in one or two power supply cages in the power bay. Each cage can contain up to seven programmable power supplies and controllers. Each programmable power supply is software controlled by means of a power supply controller in the rack.


PROGRAMMABLE VOLTAGE UUT POWER SUPPLY RACK 1
USER POWER SUPPLY
PROGRAMMABLE VOLTAGE UUT POWER SUPPLY RACK 2
POWER SUPPLY CONTROLLER

Additionally, two optional fixed UUT power supplies are also available. These fixed power supplies are typically located in the power bay, or they may be located in the pin bay. All UUT power supply outputs are routed by cables to heavy duty connectors on the receiver.

The computer (or PC)

The computer controls the overall operation of the tester. GR use a Pentium-based Personal Computer (PC) with a Windows NT operating system.

The PC:

- Controls all tester functions through its run-time system.

The PC runs the test program and controls the tester hardware, as programmed in the test program:

- Contains the software needed to develop test programs automatically.
- Allows easy file and system management.
- Provides security through user login passwords.

In addition, the PC can be used for other general tasks, such as:

- Collecting statistical data, for example data on quality and repair processes.
- Running other Windows applications.
- Networking with other systems.

Chapter Five

Using the Tester

Chapter Five

Using the Tester

The two phases of testing

The in-circuit testing process consists of two major phases:

1 - Preparing the test fixture and developing the test program

During this phase, the test program is developed and the test fixture is wired. Using circuit board information typically taken from the CAD files and data from its test libraries, the tester generates the test program. At the same time, it also develops design data for building the test fixture. The combination of test program and test fixture is referred to as the test set. A test set is usually unique for each different board type.

2 - Testing the boards

During this phase, the test set is used to test boards in a manufacturing environment.

What do I need to get started?

CAD files

The CAD files are the most convenient way of creating the circuit description. Information in these files describes component types, shapes, precise positions on the board, values and tolerances, connection information, bottom or top side placement, and much more.

Schematic diagram and parts list

These reference documents are useful when creating and debugging the test set. They help you identify parts of the circuit that may require special attention during testing.

Bare board and loaded board

Having access to a board can be helpful when designing the test fixture. For example, it can help you identify possible access problems on the board, what type of nail head would be best suited for a particular node, etc.

Test fixture kit

You need the test fixture kit only if you are manufacturing the fixture. The kit contains all the hardware for assembling and wiring the fixture. In many cases, the fixture is outsourced to a subcontractor who specializes in building fixtures.

Anything else I need before getting started?

Yes. You need to develop a TEST STRATEGY.

Start by asking yourself some questions about how you want various parts of the board to be tested. Also understand the testers capabilities.

What types of faults do you want to detect and what types of tests do you want to run?

What type of faults do you want the tester to detect: opens, shorts, incorrect parts, out-of-tolerance analog components, etc. Will you be testing digital components, functionally testing analog circuits, or making timing measurements? Will you need to perform hybrid tests, or have to deal with Programmable-Array Logic (PALs)? Will you run vectorless tests (described in Chapter 6) which may require proper access and special fixturing hardware if you're using GenRad's Opens Xpress™ software tool.

Is the tester capable of performing these tests?

Does the tester have a sufficient pin count to test the board? Will the standard instrumentation hardware be sufficient? Does it have sufficient UUT power supplies? Will the testing need to be performed on more than one type of system. Low-end testers, for example, may not have digital testing capabilities.

How soon is the test set needed?

Scheduling considerations can affect the extent of the testing. You may need to scale down the test program initially, then add to it as time permits.

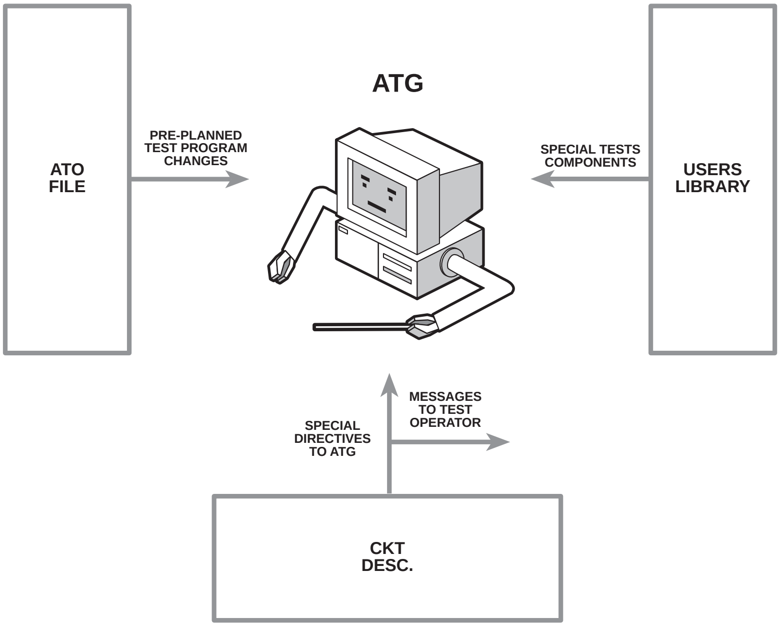
What is the desired throughput?

How long will it take to test a passing (good) board? Will you do panel testing or use a split fixture? Will you trade off testing some components for a faster test time?

How can I specify special testing requirements?

GenRad testers have several features designed to help you take care of those issues. For example:

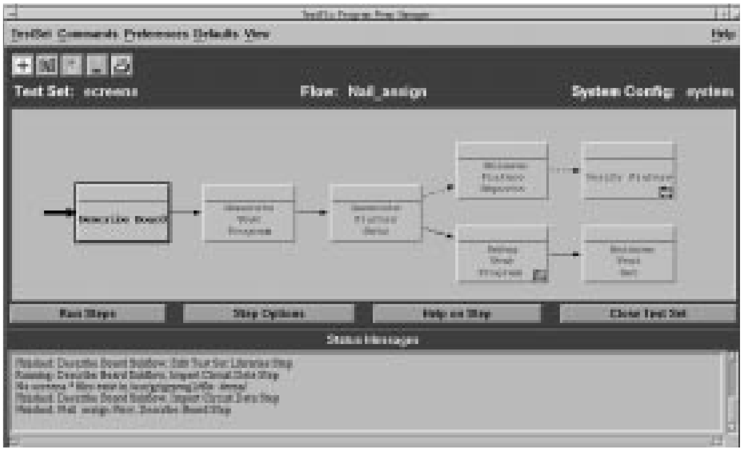
- You can insert special directives (called flagspecs) in the circuit description, which can either guide the Automatic Test Generation (ATG) process in developing the test program or can provide special instructions to the operator during testing.
- You can create an Automatic Test Options (ATO) file that will make pre-planned changes to the test program that ATG develops.
- You can create or import new tests or component descriptions for the ATG User's Library.


ATG
ATO FILE
PRE-PLANNED TEST PROGRAM CHANGES
SPECIAL TESTS COMPONENTS
USERS LIBRARY
SPECIAL DIRECTIVES TO ATG
MESSAGES TO TEST OPERATOR
CKT DESC.

Now, how do I start to develop the test set?

GenRad's TestFlo™ can help guide you through that process.

Generating new test programs for complex PCBs can be challenging and time-consuming for the test engineer. To simplify this process, GenRad has created a set of software tools with a graphic user interface, called TestFlo. As shown below, the top-level TestFlo screen displays an icon for each major step associated with developing the test set.



Test Set: screens

Flow: Nail_assign

System Config: system

In addition to providing a wide variety of software tools to simplify the job, TestFlo also shows the dependencies between the major steps along the way. It ensures that all necessary steps are performed in the correct sequence, and lets you monitor the progress.

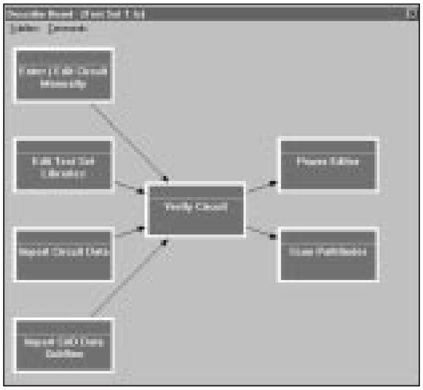
TestFlo provides tools that eliminate many of the test programming activities that were traditionally performed manually. Examples of these tools are:

- **Power Supply Editor** - Allows you to describe the power supply requirements of the UUT quickly and easily. Based on the requirements you enter, the Power Supply Editor automatically generates test program statements to power-up, power-down, and discharge the UUT power nodes.
- **Library Toolset** - Allows you to view the various libraries (analog, digital, hybrid, boundary scan,* CAD) and easily create, modify, and copy device tests.
- **Digital Device Models** - Can automatically be created using GenRad's Boundary Scan, Xpress Model,™ and NAND Tree software (described in Chapter 6). Xpress Model automatically generates digital device tests based on device pin and pattern information. Device pin information is entered, patterns are assigned and run against the device, and output responses are learned.

We'll see how TestFlo works as we go through the major steps in developing a test set.

The Describe Board step

The first step is to develop the circuit description. The tester's Automatic Test Generator (ATG) software uses this circuit description to produce a test program. When you click the Describe Board icon on the TestFlo screen, the following subflow is displayed.

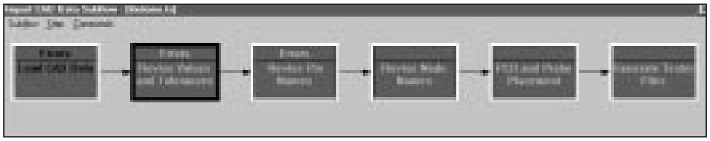


If you don't have access to CAD files, you will have to manually create a new circuit description file or modify an existing one. Clicking the Enter/Edit Circuit Manually icon lets you do this. A software tool called CKTGEN will prompt you for all the information it needs for the circuit description. Or, you can use the file editor to create a new file or modify an existing one.

The Import CAD Data Subflow

In most cases, however, the CAD files are used to automatically generate the circuit description. Simply click the Import CAD Data Subflow icon and the following subflow will be displayed.

The Import CAD Data Subflow guides you through the following steps:



- **Load CAD Data -** This step invokes a software tool called CB/Test which retrieves the complete physical database of the board from the CAD files. Based on the CAD data, CB/Test produces various reports on fixture design which include wire lists, fixture plots, and drill tapes.
- **Edit Values and Tolerances -** This step lets you review and edit the CAD data using a convenient spreadsheet format.

- **Edit Pin Names and Node Names** - This step lets you review and edit the pin names and node names in the same way you edited values and tolerances in the previous step.
- **Probe Placement** - This step helps with fixture development by recommending where the probes should come in contact with the circuit nodes, what size probes to use, and what nodes appear to be inaccessible.
- **Generate Tester Files** - This step outputs the circuit description files needed by ATG to generate the test program and the nail database files needed for nail assignment (which we'll look into later).

After the circuit description is complete, the Describe Board subflow shows several other steps to be considered in developing the test set:

- **Edit Test Set Libraries** - Clicking this icon invokes the Library Toolset which enables you to create or import library source files for analog components, digital tests, hybrid tests, and boundary scan components.
- **Verify Circuit** - This step verifies that a circuit description file exists, checks the file for accuracy and proper format, and reports any problems that might exist with the file. It also checks that all parts are in the test libraries.
- **Power Supply Editor** - This step allows you to describe the power supply requirements of the UUT quickly and easily. Based on the requirements you enter, the Power Supply Editor generates test program statements to power-up, power-down, and discharge the UUT power nodes. The Power Supply Editor also produces a Fixture Wiring Instructions file that informs the fixture assembler on how to wire the power supplies in the test fixture.

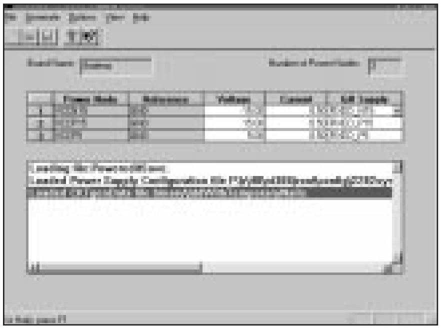
The Generate Test Program step

After creating the circuit description, the next step is to develop the test program and the test fixture. We'll start with the test program.

Clicking the Generate Test Program icon on the TestFlo screen invokes a software tool called ATG - Automatic Test Generator, which automatically develops the test program. While ATG works on the test program, TestFlo monitors the test program development and reports any problems, errors, or warnings that were encountered in the process.

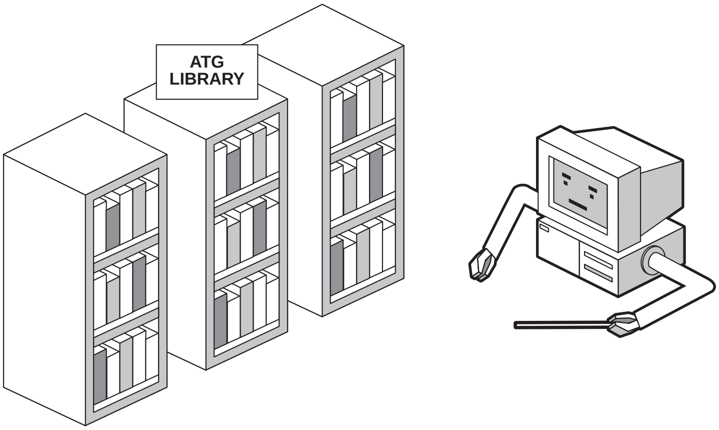
Now, more about ATG

Even with a brief introduction to in-circuit testing presented here, you can appreciate the effort required to develop the test program for a large PCB containing hundreds of nodes, components, tolerances, interconnections, test procedures, test limits, etc. ATG performs this function automatically.

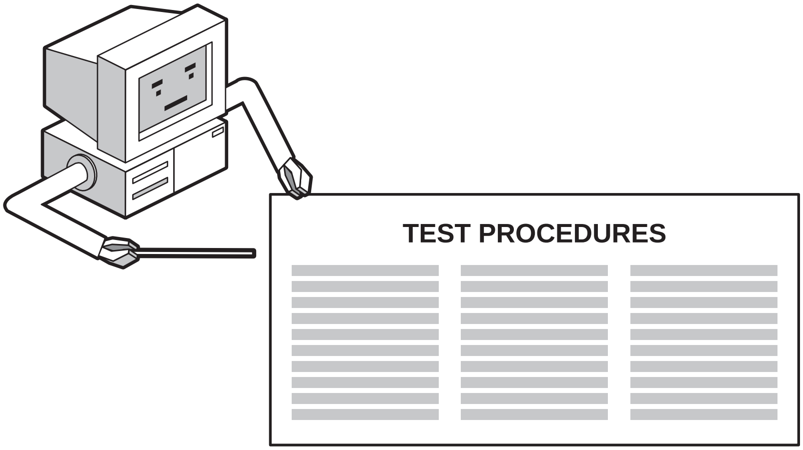


How does ATG work?

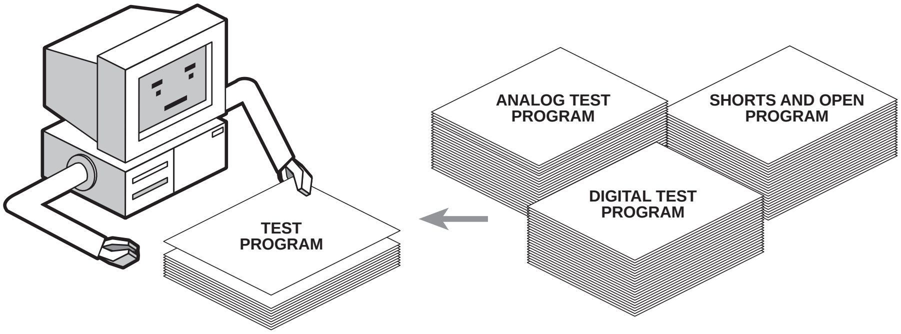
ATG uses information in the circuit description (component names, values, tolerances, interconnections, etc.) to find a test for each component on the board. ATG checks through the tester's extensive library of component and test procedures to find an acceptable test procedure for each component.



When an acceptable test is found, ATG defines the specific parameters (test voltages, guard points, delays, etc.) for customizing that general procedure to the particular in-circuit configuration of the component being tested.



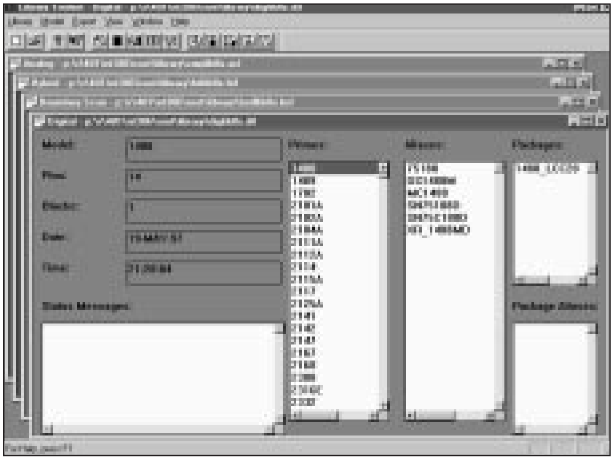
ATG then collects the individual tests and assembles them into a logical test program, (typically shorts and opens testing first, followed by analog testing, then digital testing).



The ATG libraries

In the process of developing the test program, ATG has access to several types of library files:

- **Digital Test Library** - Contains test procedures for checking hundreds of different types of digital ICs.
- **Analog Test Library** - Contains test procedures for checking basic analog characteristics such as resistance, capacitance, inductance, transistor gain, etc.
- **Hybrid Test Library** - Contains test procedures for testing a wide variety of hybrid (analog and digital) circuits.
- **Analog Component Library** - Contains circuit descriptions for complex or unusual analog components (e.g., a special resistor pack) that would not be directly recognized by the ATG software.
- **Boundary Scan Library** - Contains descriptions of boundary scan testability circuitry for boundary scan components.



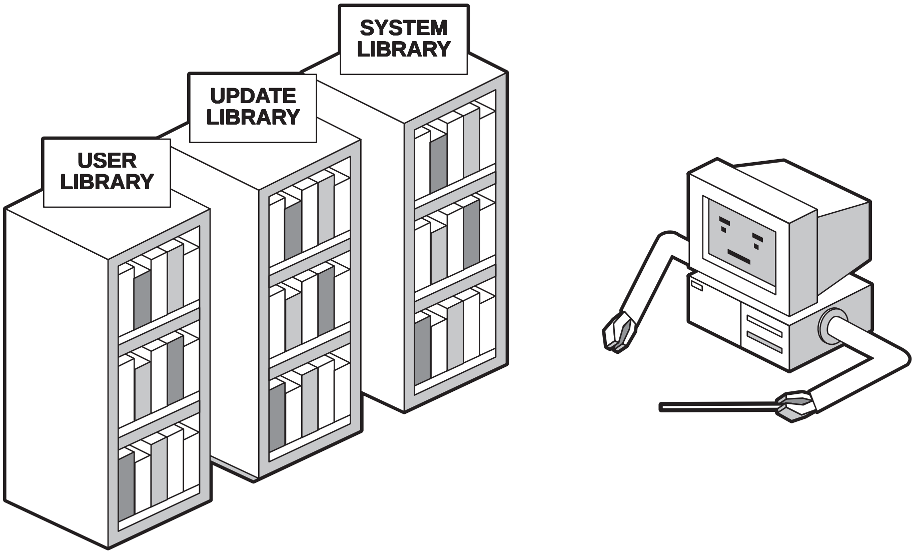
Can I change the ATG libraries?

You can customize and update the library procedures by creating your own user library files, but you cannot change the contents of the system library. You can do this by using the Library Toolset which is accessible from TestFlo's Edit Test Set Libraries icon we saw earlier in this chapter.

ATG will always search through the user library for a test procedure before it searches through the system library. If an acceptable test is found in the user library, ATG will use that test. If no acceptable test is found in the user library, then ATG will try to find one in the system library.

The assumption is that all test procedures and component descriptions that you put into your user library were put there to supersede corresponding procedures and descriptions in the system library.

In addition, GenRad testers also have an update library that allows GenRad to update system libraries in the field, on a periodic basis. With an update library in place in a user's system, ATG will always search through this update library before searching through the system library. New tests or replacement tests in this update library will, therefore, automatically appear in the test program.



How ATG develops the test program

Based on the circuit description of the board and available test procedures in the libraries, ATG attempts to find a suitable test for each component on the board. The accumulation of all these individual tests forms the test program.

ATG USES

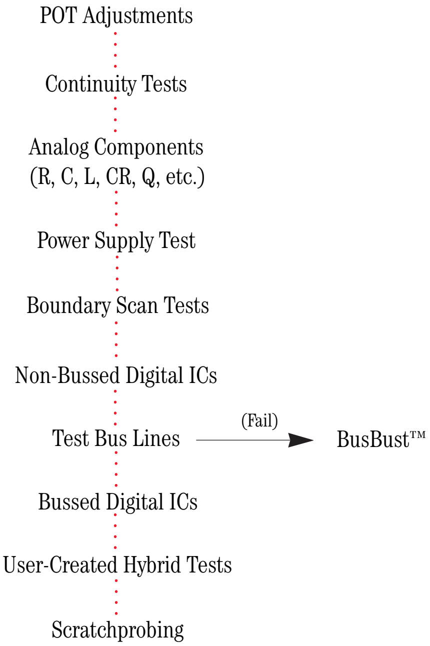
Circuit Description
Libraries
UUT Power Program
Automatic Test Opts

ATG

TO GET

Shorts and opens tests
Analog tests
Digital tests
Hybrid tests
Boundary scan tests
Other tests
Reports

The testing sequence is, of course, important. Starting with the most basic, each test level is designed to build confidence in the next higher level of tests. The testing sequence is shown in the flow chart below.



The ATG reports

When ATG finishes assembling the test program, it reports any known program deficiencies. For example, it will indicate any components for which it could not find an acceptable test. The test engineer can then modify the program to account for these deficiencies or make other changes, as appropriate.

ATG produces the following test generation reports:

- **Analog test report** - Identifies the board and the libraries used to generate the tests. Reports connectivity results (opens and shorts tests) followed by test information for each type of analog component (capacitors, transistors, resistors, etc.). Also produces a summary report rating the quality of the tests.
- **Digital test report** - Identifies the board and the libraries, then summarizes the digital test statistics and problem areas for testing that board. Also contains test information for each digital IC on the board.
- **Hybrid test report** - Summarizes the test statistics for all tests retrieved from the hybrid test library and provides detailed fault coverage for each component to show the test engineer which pins are being tested for faults.
- **Boundary Scan* configuration and fault reports** - Describes fault coverage for each Boundary Scan component that had a test created. It also reports any errors or warnings encountered during test generation.

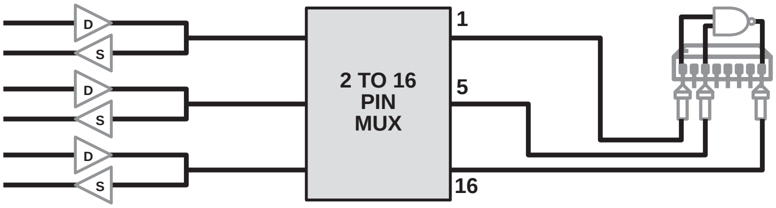
The Generate Fixture Data step

The next major step shown on the TestFlo screen is Generate Fixture Data. This step generates most of the reports needed to produce the UUT fixture. Also during this step, the tester performs an all-important nail assignment task.

What is ‘nail assignment’?

Remember in chapter 3, we saw that the digital driver/sensors are multiplexed. Each test nail does not have its own dedicated driver/sensor. Therefore, the tester cannot arbitrarily assign just any test nail to any circuit node.

Look at the following example of a driver/sensor multiplexer.



There are 2 driver/sensors available for 16 nails. If you connect 3 nails (from the same group of 16) to the NAND gate, the system cannot possibly drive both inputs and sense the output simultaneously, with only 2 driver/sensors.

To prevent this conflict, at least 1 of those 3 nails must come from another group of 16.

Do I have to worry about this?

Absolutely not!

The tester takes care of the entire nail assignment task for you. The nail assignment software analyzes the test program and other input files and automatically assigns nail numbers so that testing conflicts will not occur.

---

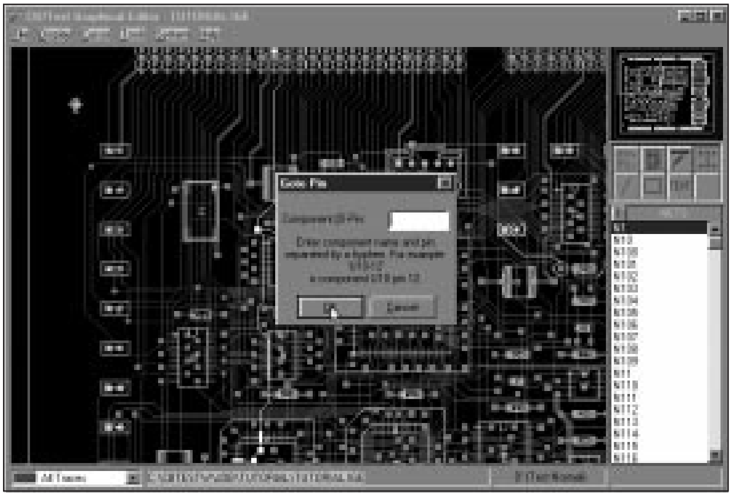
*Refer to Chapter 6 for more information on Boundary Scan.

In addition to performing the nail assignment function, this step also performs the following tasks:

- It produces a Fixture Test Program, which can be used to verify the fixture.
- It translates the Fixture Test Program and the Board Test Program into machine code so that they can be run at high speed on the tester.

Probe Placement

Another important piece of data that is useful in building the fixture is the report from the Probe Placement step, which we've already seen as part of the Describe Board subflow. The Probe Placement report shows the physical location on the circuit board where each test probe should be placed. An example is shown below.



The Release Fixture Reports step

This is an optional step that lets you extract all the files that the fixture developer needs to build your fixture. Different files are released depending on the test strategy you chose to use for this UUT.

Some of the available files include:

- **Nail assignment report -** Specifies the connection or signal path between a specific test nail and a given node, and lists all the leads connected to that node.
- **Nail fixture report -** Specifies connections between a test nail and a specific component lead at a given node. The report also describes how to wire the fixture.
- **Power supply wiring -** Describes connections between the user power supplies and the designated power block on the receiver.
- **Device probe report -** Describes the interconnection between Opens Xpress or external fixture probes and the system.
- **Nail database report -** Shows the mapping of temporary nails to target nails. Allows you to force certain temporary nails to be assigned to specific receiver nails.
- **Files produced by CB Test -** Fixture files that can be generated to meet the specific requirements for over 20 individual fixture vendors. Organizes raw fixture data from a variety of CAD and test set files to meet the criteria of fixture houses for fixture drilling and wiring.

The Verify Fixture step

This is an optional step that lets you verify the Fixture Test Program and UUT Power Test Program (produced by the Power Supply editor). This step uses the tester's run-time system and the tester hardware.

Fixture tests generated by this TestFlo step include:

- An opens test which requires a shorting plate mounted on the fixture
- A shorts test that is run with nothing mounted on the fixture

These tests check every test nail specified in the circuit description against a common point.

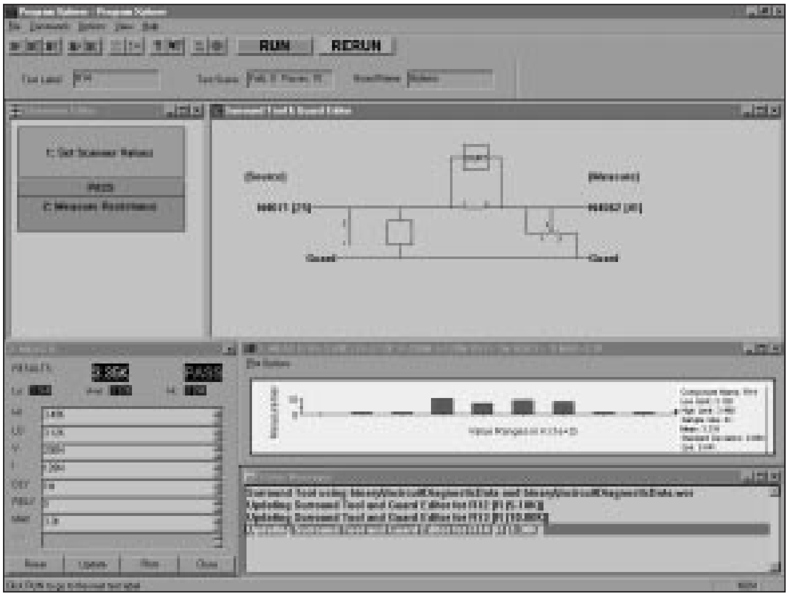
You can further test the fixture and confirm the wiring to a particular nail by using a system probe. Touching the node with the probe displays the nail and other related information.

Also available is a quicker but less thorough probing procedure in which the tester stops at each IC and prompts you to probe the IC pins.

The Debug Test Program step

Like any other new piece of software, the test program typically needs some debugging and fine-tuning. The tester has software that helps you perform that task.

Start by mounting the new test fixture on the system, place a known good board on the fixture, and click on the Debug Test Program icon. The tester uses its runtime system and test hardware to run the test program.



The tester has an interactive debug mode which displays the test results and allows you to analyze the displayed data, instruct the test system to change test statements, and immediately see the results of these changes by re-running the test.

However, the software also includes an automatic debug mode and several other debugging tools.

Tools for debugging the test program

- **Autodebug -** This tool takes an ATG-generated program, runs it on a verified test fixture, and attempts to greenlight* as many failing component tests as possible without operator intervention. For example, if an analog test fails, Autodebug will modify delays and guard points (within prescribed limits) in an attempt to greenlight that test. Autodebug records its activity in a log file so that you can see what changes were made to the test program. The log file also flags tests that require further action.
- **Program Xplorer -** This powerful set of tools speeds up debug by providing a real-time environment that includes schematics, charts, and graphical panels. You are able to make adjustments to the test program and immediately see the results.

---

*The term greenlighting is often used to indicate the successful testing of a board or a component.

- **Hypertext Help -** Online help is available to aid you in debugging a wide variety of component tests. For example, if you wanted information on debugging a transistor test, the hypertext help file on that subject would explain how the test is performed, show an example of a failed transistor test, describe each test statement, present various debugging techniques, etc.
- **Digital Waveform Display -** This tool displays a graphical representation of driver and sensor states, similar to a logic analyzer. It also includes information identifying the start and end of loops and subroutines, signal names of each pin, pin numbers, etc. Logic states differing from their expected values are highlighted, allowing you to quickly locate failing pins.
- **Learn Mode -** This mode lets the tester automatically learn the state of all device outputs that were specified as 'unknown' in the test program. Learn mode is typically used to learn output states of devices that cannot be easily determined in any other way, such as PROMs and ROMs. As the test program is run on a verified circuit board, the tester observes and learns the unknown outputs from known good devices. It then uses this learned information to update the unknown states in the pin memories to the states newly learned during an actual test.
- **Electronic Books -** GenRad has an integrated browser to read the software manuals online. This new capability allows you to view product documentation online, search for information across different manual sets, cut and paste into an editor, annotate the manuals with your comments, etc.

Fault grading

By determining in advance which faults are covered by the test program and which faults are not, you can grade the quality of the test program before going into production. The tester's ALLFAULT software can do this for you by measuring and evaluating analog tests and by emulating digital hardware faults.

For digital devices, ALLFAULT automatically inserts all possible input stuck-at-zero and stuck-at-one faults to verify that the test program will propagate these failures to the outputs. Through the same process, all output failure conditions can be tested. The result is a high confidence that the test program will not propagate unwanted failures to later states of the process.

For analog devices, ALLFAULT runs the test multiple times and flags those components that have measurements that are unstable and require additional debugging.

When finished, ALLFAULT issues a report that identifies digital tests that have poor fault coverage, and analog tests that are unstable or close to a measurement limit. This information allows you to manage the test development process for the highest fault coverage.

Data logging

The data logging option enables the tester to collect all test measurements for future analysis. You can use the Data Display tool (described later in this chapter) to analyze the log results and generate graphical reports.

The Release Test Set step

This step allows you to release debugged test programs to the manufacturing floor.

Running the test program

Once the test set is released to production, the board testing process begins and the tester starts to earn its keep. From here on, the board testing phase becomes a repetitive procedure, consisting of a few simple steps. At the start of a testing session, the operator checks the instructions written for that test set.

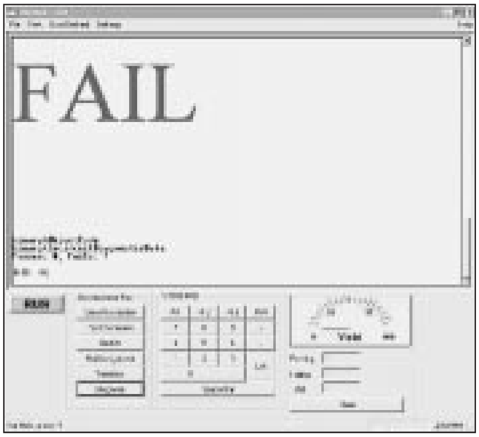
The operator then mounts the fixture on the system,

places the PCB on the fixture,

and runs the test program.

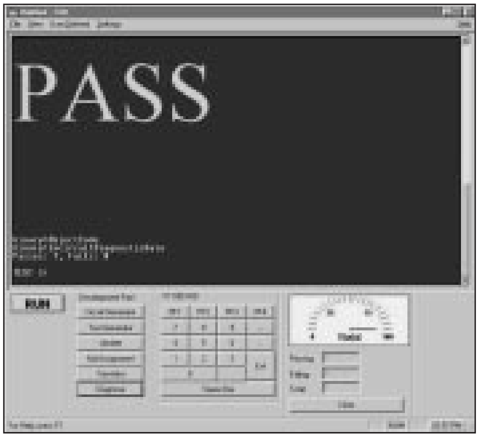
Diagnosing the test results

If the board fails, the operator performs whatever diagnostic procedures are called for by the displayed messages. For example, potentiometers may need to be adjusted or switches may need to be set.



FAIL

After a test run is complete, system-generated diagnostic messages can be attached to bad boards and forwarded to a repair station. Good boards can be forwarded to further assembly.



PASS

Continuing the test programming effort

It is important to monitor the production testing phase to make sure that the test program really finds all the faults that it is supposed to. If it doesn't, you should analyze the failures detected at the next test stage (system test) and use the data to modify the test program, as needed. You can also use the tester software to plot the distribution of test results for several boards. This will show how stable the analog measurements are.

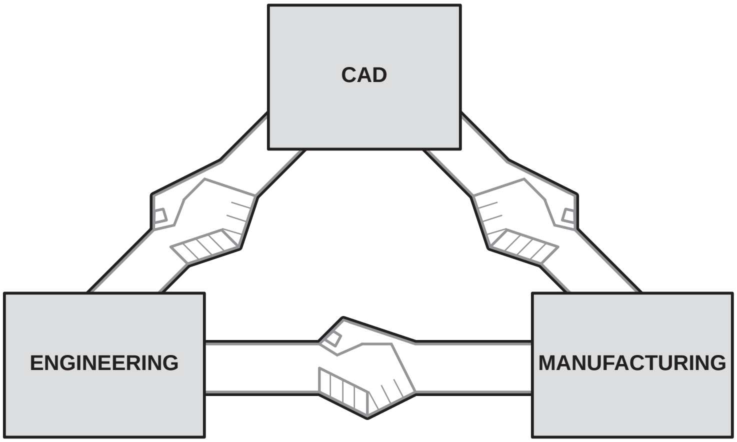
Data Display Tool

The tester also has data display software that makes it easy to display, analyze, and print process information. The software extracts performance statistics from the tester's log file. It also identifies trends or deficiencies in particular components by analyzing the failures that occur when you test a board.

All you have to do is import the files you want to analyze and decide the type of report you want to generate. After the report is generated, you can add text to further clarify the report before printing it.

An on-going effort

It's a good idea to continue working closely with the other test departments monitoring test results until all problems are resolved.



Chapter Six

Techniques For Improving Fault Coverage

Chapter Six

Techniques For Improving Fault Coverage

Overview

With certain PCB production goals to attain (i.e., yield, fault coverage throughput, beat-rate of the line, etc.), you need to understand the types of PCB faults occurring in your process. Up to now, we've examined the defects caused by the PCB manufacturing process and how in-circuit test and inspection detects most of those process faults (opens, shorts, missing or misoriented components, etc). Today's manufacturing environment presents you with time-to-market and time-to-volume pressures. This means that you have a shorter product ramp-up period. Your test strategy becomes even more important since you will have less time to develop your test but must have fault coverage regardless.

This chapter introduces you to some additional test techniques which can help you develop proper test strategies for your particular needs. You will also get a glimpse at some features of the in-circuit tester that go beyond just simple testing and can help streamline your PCB manufacturing process.

The chapter covers three major areas of testing:

- **Unpowered, vectorless testing** - Test situations where you do not have to power up the board and do not have to develop test vectors.
- **Components with built-in testability** - Testing components that have test structures built into them (boundary scan, NAND tree).
- **Function Testing** - Tests that check the functional operation of a circuit or group of circuits. This includes hybrid testing, cluster testing, bus emulation, memory testing, and other tests.

A few words about 'vectorless' testing

We've seen how a bareboard can pick up various manufacturing defects as it moves through the assembly line. Do you know what one of the most common process defects is? Open solder joints. This is especially true with boards that contain surface mount devices. Detecting open solder joints on digital devices by using test vectors requires high quality and reliable test vectors that may not be readily available.

As PCBs have become larger and more complex, the allowable ramp-up time has actually become shorter. As a result, you may not have the time to create a complete set of test vectors for non-standard devices such as ASICs (Application-Specific Integrated Circuit). To add to the problem, the pin configuration of some complex ICs do not provide much access for testing the device anyway.

The good news is that semiconductor manufacturers have greatly reduced the number of faulty components shipped to PCB manufacturers. In many cases, this lets you get by without functionally testing every digital device during in-circuit testing. Therefore, you may not need test vectors for certain complex devices.

The additional good news is that several test techniques have been developed to target open solder joints - the most common source of failure on digital ICs. These techniques do not verify the operational integrity of the device. However, they can pinpoint the exact open pin that is causing a failure and they do not require test vectors.

In preparing your test strategy, you need to ask yourself several questions on the testability of the board you're testing. For example:

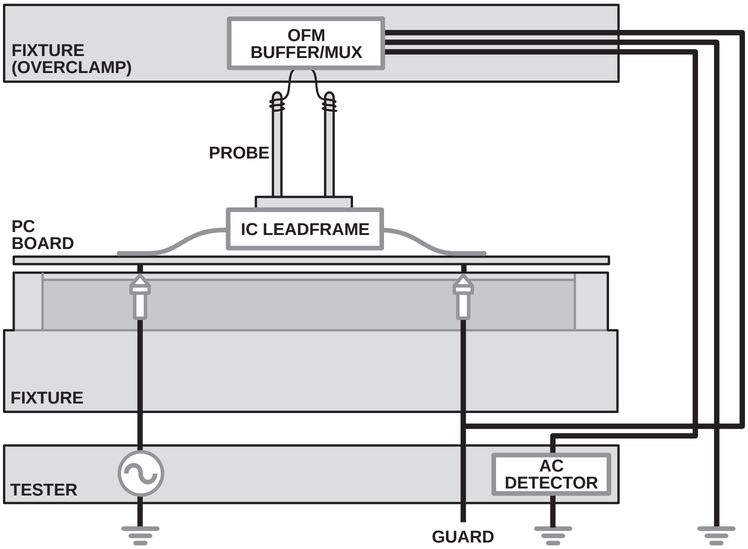
- Can you access all the nodes?
- Do you want to apply power to the board during testing?
- Do you have test vectors for all components?
- Does the board have components with built-in test features?

The answers to these questions will determine which test techniques described in this chapter will be most applicable to your particular test strategy.

Unpowered opens testing

How would you test non-standard devices such as ASICs (Application-Specific Integrated Circuit) which traditionally take a lot of test development time? Or, what if you had to test for opens on RS-232 or SIMM connectors - tests that are not usually performed until system test? A technique that can solve these problems is unpowered opens testing.

This technique works by sourcing an AC voltage through each lead of a device, then measuring the resulting AC voltage via a probe that is attached to the test fixture. If an open exists between the circuit board and the solder lead, there will be a substantial difference between the measured voltage and the expected capacitance.



This test technique has many advantages:

- Test procedures are created automatically from the circuit description of the board.
- Device libraries are not required.
- Works on all IC types, on both through-hole and surface-mount components, on any package size and lead density, and most connector types.
- Works when full access is not available.
- Provides fault diagnostics to the open device pin.

Analog harmonic testing

The unpowered opens test described on the previous page cannot be used to test IC devices that shield signal propagation to a probe, or devices with ground planes.

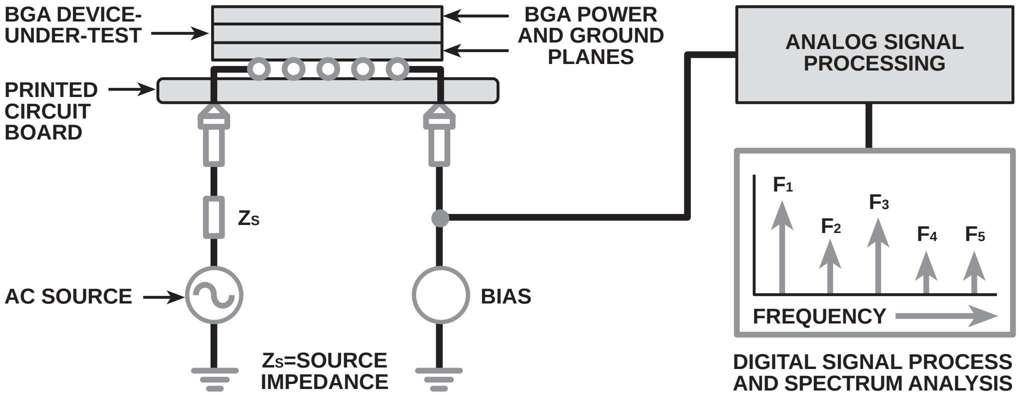
'Analog harmonic testing' is an elegant term that describes a vectorless test method that can be used to locate opens and marginal solder connections on such devices. As the name implies, it uses the harmonic signatures of the integrated circuit devices.

The technique uses three pins on an IC to check for an open on a specific pin:

- The source pin is the pin being checked for an open connection.
- A second pin is called the detector pin and is used to measure the response signal.
- The third pin is the common ground connection.

The technique works as follows:

- The AC source applies a signal through the test nail into the device.
- The AC measurement devices measure the response at the detector pin.
- Software calculates the second harmonic of the response voltage.
- An open or marginal solder joint typically displays a 1-to-10 difference in response value from a good solder joint.



Design-for-testability

The term ‘design-for-testability’ refers to the ongoing effort by both component and board designers to improve the observation and control of their designs during test. This chapter describes two design-for-testability methods that have been developed for digital ICs: boundary scan and NAND tree.

Devices designed for testability simplify the developing of tests to detect shorts and opens on PCBs when dealing with issues such as limited access, unavailable test vectors, and device isolation.

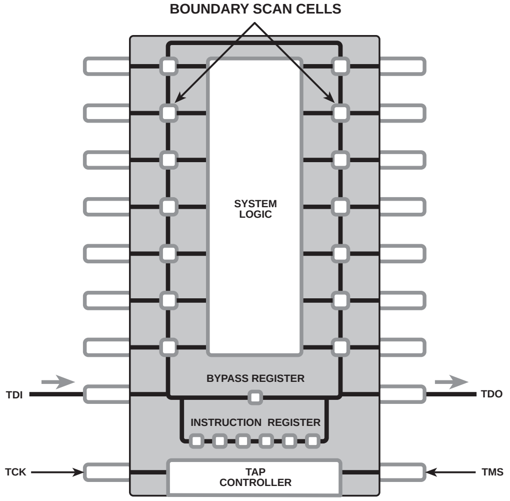
Boundary scan testing

Suppose you have just begun to develop a test program for a board that will soon be into production. Looking at the schematic drawings and device data sheets, you see that several ICs are boundary-scan compliant. What does that mean? And more importantly, can it help you to increase overall fault coverage and reduce test generation time?

What is boundary scan?

Boundary scan provides a way to put test electronics inside every pin in a component. The term ‘boundary’ refers to the location of the test circuitry - between the component’s internal circuitry and the input and output pins. The term ‘scan’ refers to the type of test circuitry inside the component - a serial shift register often referred to as a ‘scan path’. Boundary scan components provide access to this scan path via four or five dedicated pins.

Boundary scan components allow the tester to control and observe the component’s scan path. This scan path is connected between the component’s TDI and TDO pins. The path consists of a series of boundary scan cells, which interconnect to form a shift register. There is one cell for each pair of input and output pins.



Each boundary scan component contains registers that provide the following functions:

- An Instruction Register specifies the actions to be performed by the component’s internal scan register.
- A Boundary Scan Register controls and observes the component’s I/O pins.
- A Bypass Register can shorten the scan path between the TDI and TDO to one cell in the internal shift register.

The tester can perform pin-level diagnostics by supplying both a logic 1 and a logic 0 to the input of a boundary scan device, capturing it into a scan cell, and shifting it through the scan path for comparison against known good data.

Benefits of boundary scan

Some of the benefits of boundary scan testing include:

- Automatic test vector generation to detect opens and shorts on device pins
- Fast test development
- Excellent diagnostics
- Full access to nodes is not required
- Option to verify correct device with IDCODE instruction and correct operation with RUNBIST instruction.

Refer to the GenRad handbook Meeting the Challenge of Boundary Scan for more information on boundary scan testing.

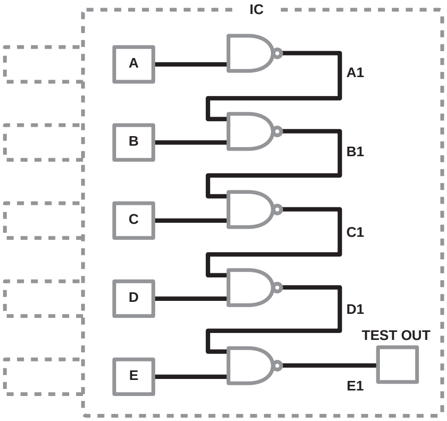
NAND tree testing

In addition to boundary-scan components, vendors may include other test structures inside their ICs to help test the device. One such feature is a NAND tree. Using the circuits' internal NAND trees, in-circuit tests can be developed to check for open input and bidirectional pins, as well as stuck-at-0 and stuck-at-1 input pins.

Here's how it works.

Each device input is connected to one input of a NAND gate. The other input to the NAND gate is from the input pin directly up the tree.

The test systematically toggles each input pin of the NAND tree. The test starts by holding all inputs low, thereby enabling the NAND tree. Starting from the gate closest to the output, the input is changed from a low to a high, thus causing the output to change to a low.



Every time the test moves one branch up the tree and applies a high input, the output should also toggle.

Note that this test technique needs full access to the input pins. Also, it cannot test output pins. However, it can be implemented with little programming and may complement another test in your strategy.

Function testing

Function testing allows the tester to detect faults related to the incorrect operation of individual UUT components and groups of components. This includes hybrid testing, cluster testing, bus emulation and memory testing.

Hybrid testing

So far in this book, we've examined some basic techniques used for testing simple analog and digital devices such as resistors, capacitors, diodes, and NAND gates. Examples were used to illustrate the techniques. However, most of these device tests are already stored in the tester library and do not have to be created again.

But, how do you handle more complex devices for which there are no library tests?

Designers are continually building more functionality into new devices. Product manufacturers approve this concept because it reduces board size and materials cost, and improves product performance. Test engineers, however, are faced with greater challenges. They must develop tests to check complex analog, digital, or mixed-signal functions within these devices. Examples of these more complex devices include:

- Analog devices such as oscillators, filters, window detectors
- Digital devices such as function generators and DSP devices
- Mixed signal devices such as digital-to-analog converters

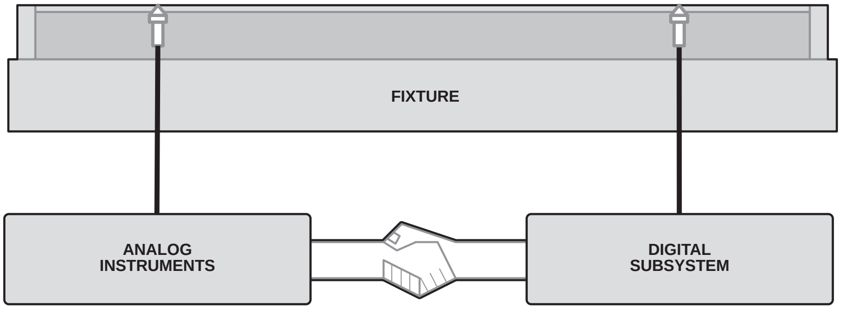
The rest of this chapter addresses these problems and shows examples of how you can deal with them.

For example, one test technique described in this chapter is cluster testing. Sometimes you may find it convenient to group components together into a functional circuit (such as a low-pass filter) and test those components as a cluster.

What to consider when testing complex devices

When you encounter complex devices that have no tests in the tester libraries, you need to consider the capabilities of the tester in order to know how to deal with those devices. Here are three important factors:

- You must be able to use a variety of internal system test instruments and probably some external instruments.
- The tester must be capable of automatic test program generation or flexible coding to control or include routines that will process and display data arrays.
- You must be able to precisely synchronize the timing between analog test instruments and digital bursts.

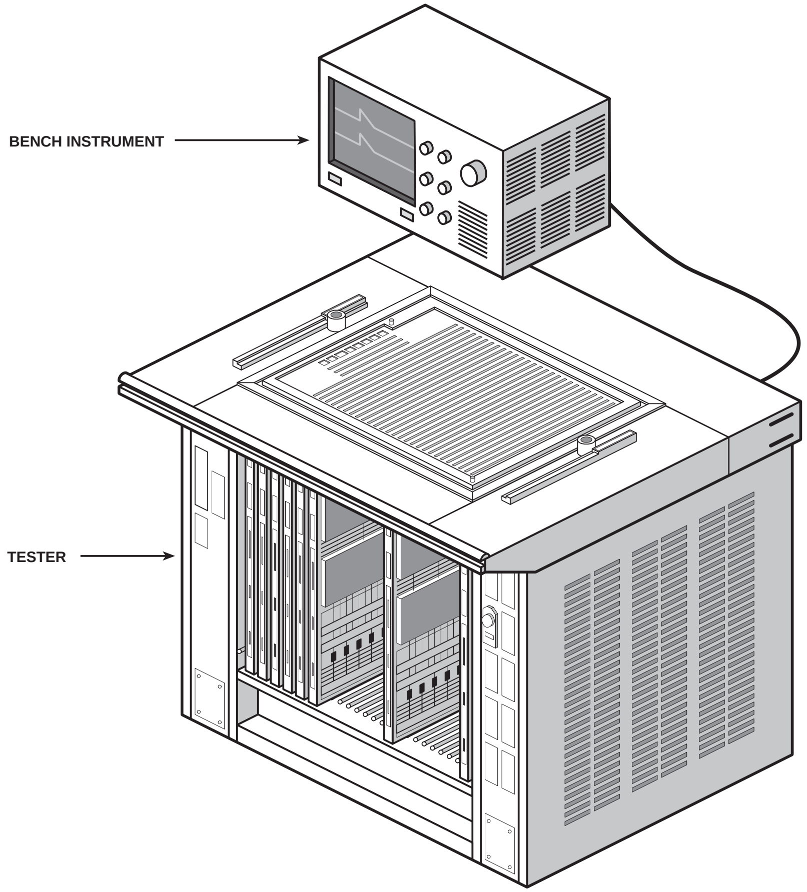


Tell me more about test hardware and test development

System hardware - The standard analog test instruments introduced in Chapter 3 included voltage and current sources, voltmeters, current meters, etc. Additional test instruments are available that can expand the capabilities of these standard instruments and provide altogether new functions. Here are some of them:

- **Arbitrary waveform generator** - Provides various stimuli to the circuit (sine, square, triangular, or user-defined waveforms).
- **Digital Multimeter** - Performs voltage and current measurements. Contains an analog-to-digital converter and instrument memory to sample and digitize analog signals.
- **Frequency/Time Meter** - Measures frequency, period, and time intervals.

You can use these test instruments independently of the digital subsystem or you can synchronize each of them to an event in a digital burst. For example, you can start a measurement, enable a source, reprogram a source's frequency, all with the timing precision of the digital subsystem.



Can I use IEEE-488 or VXI instruments for hybrid testing?

You can. In fact, most in-circuit or combinational testers typically require that you use IEEE-488 or VXI-controlled test instruments for hybrid testing.

There are many versatile and accurate IEEE-488 and VXI instruments available, BUT, they are not without drawbacks. GenRad has software (Hybrid Test Library and Hybrid Test Generator) that can simplify the programming of these external instruments.

How about hybrid test development?

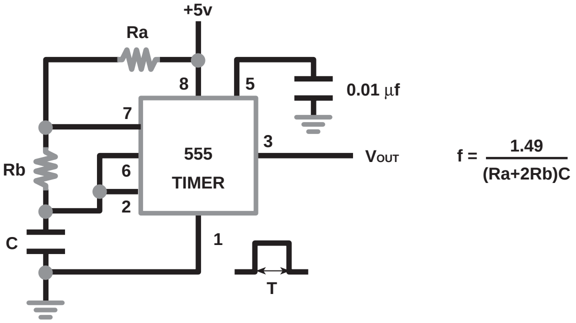
Many hybrid tests can be found in the hybrid test library. If you need to create a test, you can use any of the internal or external hardware resources available and the system's flexible switching matrix. Once a test is developed, you can include it in the test library for future use.

Measuring frequency and pulse width

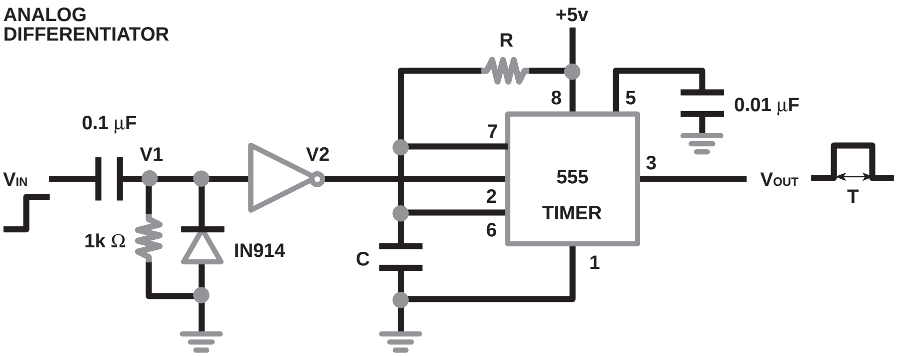
Suppose you had to test the frequency of an oscillator or the pulse width of a one-shot. None of the test instruments presented in Chapter 3 can measure frequency or pulse width - two very common measurements needed to fully test a PCB.

The Frequency/Time Meter (FTM), however, fills that need. The FTM connects to a test nail in much the same way as the standard dc source and measure instruments, except that the FTM uses its own switching matrix.

- The following example shows an oscillator circuit. When power is applied to the UUT, the circuit produces an oscillating output signal. The circuit is tested by measuring the frequency of the output signal to ensure that it is within tolerance.



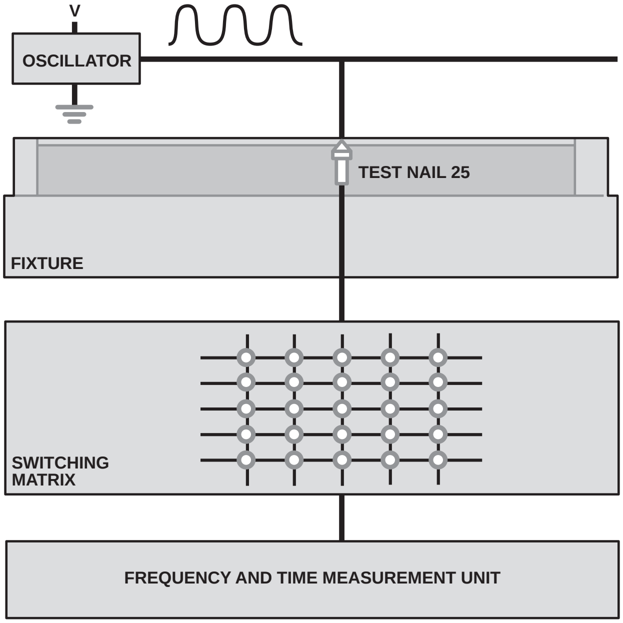
The second example is a one-shot circuit. When the one-shot is triggered, it produces a single output pulse. The width of the pulse is determined by a resistor-capacitor network in the circuit. The circuit is tested by measuring pulse width to ensure that it is within tolerance.



Testing high-speed critical signals

Suppose the oscillator output in the previous example was a fairly high frequency (say, 50 MHz or more), which had to be measured very accurately. The FTM can handle that frequency but a standard switching matrix can cause a problem. The signal path in a standard switching matrix may not be able to pass such a high frequency signal without introducing some attenuation or distortion. This would cause the tester to measure a distorted signal and report an inaccurate frequency value for the oscillator output.

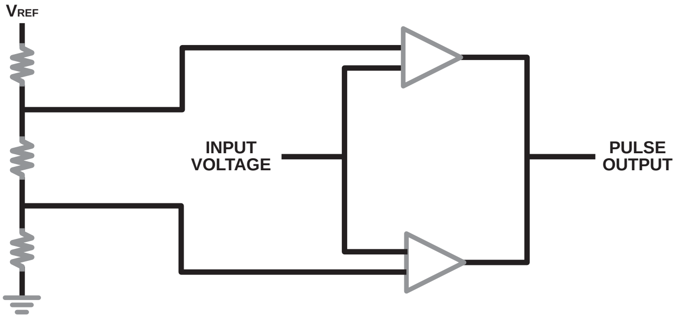
To handle these situations, the tester has a High-Frequency Scanner. Although not strictly a test instrument, this unit provides a high-frequency alternate path between the test nail and the test instrument. In the example shown below, the high-frequency scanner connects the oscillator output (the critical signal) to the FTM unit. This allows the signal to reach the FTM unit without distortion, resulting in an accurate measurement of the signal frequency.

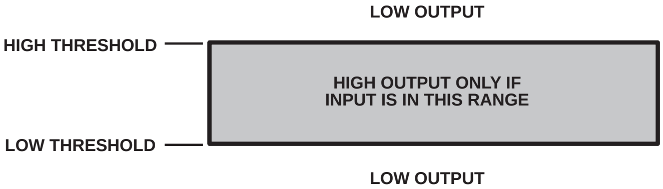


Now for something more complex - a window detector circuit

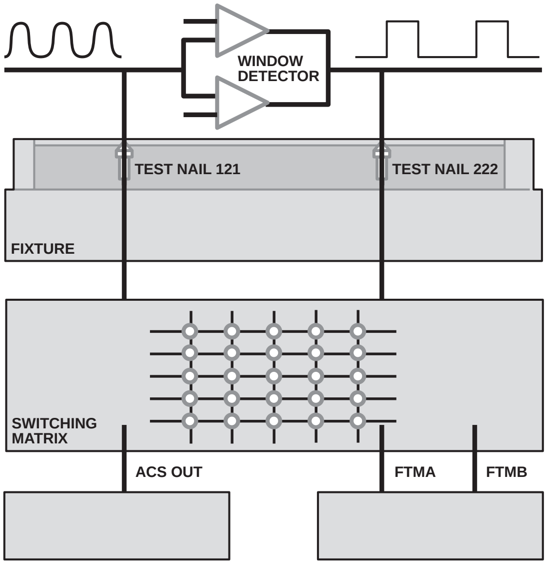
What if the circuit you’re testing requires both analog and digital test instruments? Here’s an example of how you can use both the AC source and the Frequency/Time Meter (FTM) to test such a circuit dynamically.

The circuit is a window detector, which works as follows. The resistor network sets an upper and lower threshold. When the input signal is between these two thresholds, the output of the circuit is high. When the input signal is outside of these thresholds, the output is low.



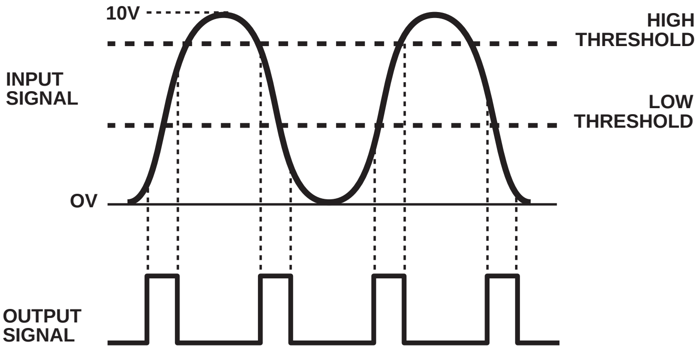


The AC source and FTM are connected to the window detector circuit as shown below. The AC source applies a 5V, 500 Hz, sine wave input to the circuit. The FTM measures the frequency and pulse width of the output signal.



As shown in the timing diagram below, the applied sine wave is offset from 0V so that the input voltage to the circuit is always positive. Also, the amplitude of the sine wave is set so that the applied voltage will periodically exceed both thresholds of the circuit. As a result, during each cycle of the sine wave, the input voltage crosses the two thresholds twice (once on the way up and once on the way down). This produces two output pulses per cycle, one each time the input passes between the thresholds.

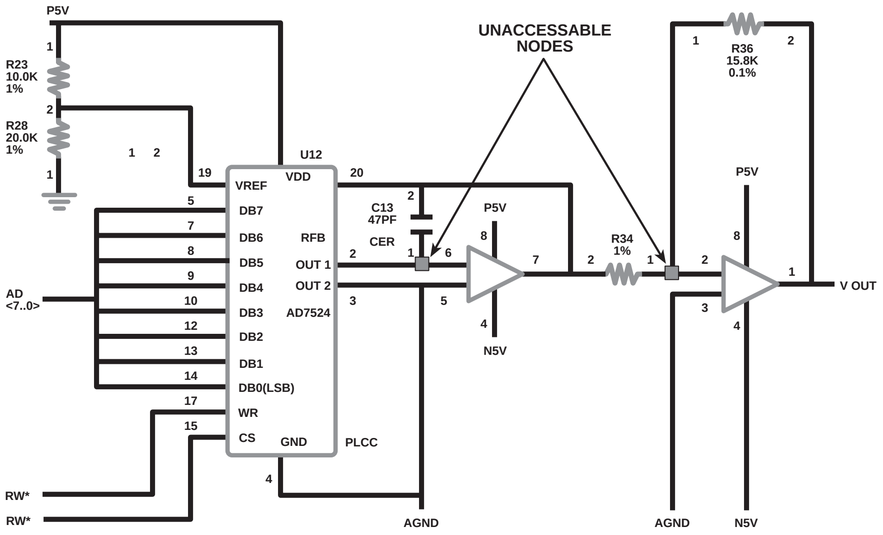
While a static measurement might be made using the DC source and measure units, this simple dynamic test provides a more extensive view of the circuit operation.



Cluster testing

When creating the circuit description for a PCB or when later viewing the fixture report, you may find that some nodes on the board are not accessible. What can you do?

One approach is for you to define a circuit cluster that includes the inaccessible nodes. A cluster is a group of UUT components and component pins that the tester will test as one unit. In so doing, the tester will be able to determine the functionality of the inaccessible nodes by monitoring outputs at the accessible nodes. An example of a circuit cluster is shown below.



You define a cluster by including the following information in the circuit description file:

- The name of the cluster
- The components that are in the cluster
- All component pins that are part of the cluster
- All external pins in the cluster accessible to the tester

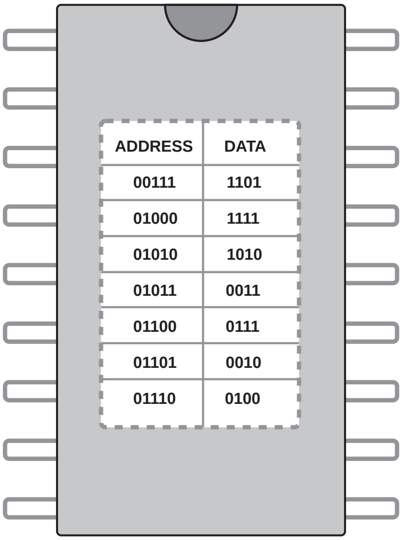
You can then create test vectors that will test the functionality of the cluster components. If the test detects a failure, you can use Guided Probe diagnostics which will pinpoint the faulty cluster device.

Full-cell memory testing

Memory testing presents its own set of unique challenges. The first phase of any memory test is to make sure that the correct memory device has been placed on the board. The next phase is to check that all the memory cells are working properly (full cell testing).

The test library contains models of many commonly used RAMs (Read/Write Memories) and ROMs (Read-Only Memories). All memory models contain standard memory parameters - number of locations, read/write timing, pin assignments, etc.

Full cell memory testing requires many more test vectors than the tester can store sequentially in Driver/Sensor (D/S) memory. Therefore, full-cell memory testing uses looping constructs and indirect addressing. Looping allows large numbers of test steps to be stored in very few D/S memory locations. Indirect addressing allows the tester to drive the address nails of the memory device efficiently to any state, using the same test statement within a loop.



ADDRESS	DATA
00111	1101
01000	1111
01010	1010
01011	0011
01100	0111
01101	0010
01110	0100

RAM testing

The ATG-developed RAM test consists of a “march” pattern, which sequentially accesses each memory location, writes ones and zeros to each cell, reads them back, and checks the results.

ROM testing

To support full cell testing of ROM devices, a Cyclic Redundancy Check (CRC) capability is designed into each D/S nail. As the memory test marches through each ROM address, the tester reads the contents of that location and sends the results into the CRC circuit. The CRC logic calculates a CRC value for the ROM under test. This CRC is compared with expected CRC values previously learned from a known-good ROM device.

The important benefit of CRC collection is that you do not have to specify the expected value of the data for each address location. Instead, all the sensed data is used to generate one CRC signature for the entire device. This approach saves thousands of test statements, resulting in a compact test.

Bank testing

You can make testing even more efficient by using bank testing. In bank testing, you use one digital burst to test multiple devices in parallel. This results in a smaller test source file and faster test execution. Two or more devices are considered to be in parallel when the devices are of the same type and can be enabled simultaneously.

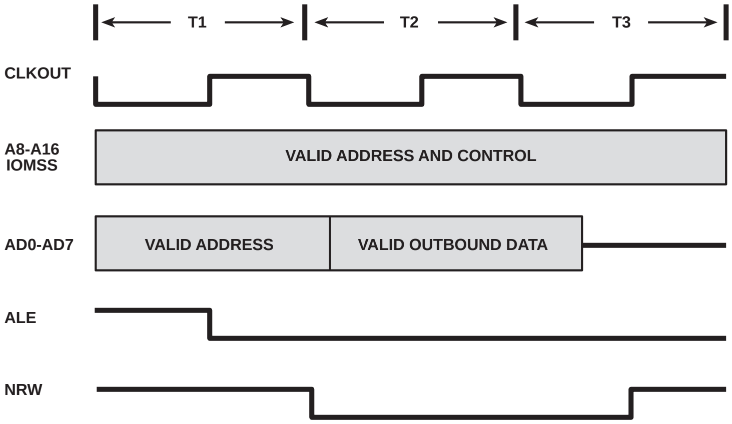
NOTE: Bank testing is not restricted to memory banks. You can use it to test any set of like devices in parallel.

Bus emulation

Suppose you needed to test a bus-structured UUT, that is, a UUT where the tester can access most of the functionality from a central bus. This structure is typical in boards that contain a processor. You can use an emulation technique, in which the tester emulates (takes the place of) some hardware or software component on the board during the test. For this example, we'll use one of these emulation techniques, called bus emulation.

During bus emulation, the tester acts like a bus master in order to write and read all of the register locations accessible from the bus.

The following diagram is an example of a bus protocol that you might find in a manufacturer's data book. You need to understand the interaction of the bus signals and how to manipulate them in order to write tests that will check if the bus works properly.



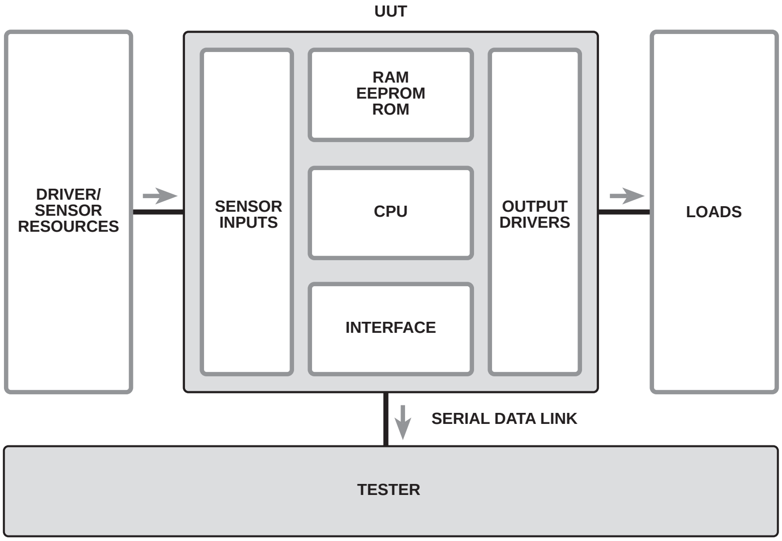
Serial communication interface testing

Some UUTs contain an RS-232 serial port that is used for development and debug purposes. This serial port can also be used to send test results to a tester. How do you test these devices?

The tester can optionally contain some serial communication instrument software, which allows it to communicate with a UUT's serial port. An example of one such available option is the Vehicle Control Interface (VCI) module. The VCI resides on a Custom Function Board. You had a brief look at this type of board back in Chapter 4.

The VCI was designed specifically to test automotive Electronic Control Units (ECUs) via the ECU's serial ports. Here's how the VCI module works.

The test is run at the in-circuit test stage of the assembly process. The Custom Function Board plugs into the tester's multiplexer, thus taking full advantage of the system's test instruments and test programs. The VCI module connects to the EC-under-test either through a scanner or directly through the test fixture.



After the tester performs the traditional in-circuit tests for shorts, opens, and assembly or component faults, it initializes the VCI for the specific communication protocol required by the manufacturer. The ECU is then placed in Diagnostic Mode and the tester sends a message to the ECU directing it to perform some action. When the ECU responds, the tester checks the response to see if the data is within limits specified by the manufacturer.

The VCI can detect various faults such as the wrong revision level of microprocessor at an early stage, before the unit moves on to full environmental functional testing. Catching and correcting errors on ECU boards before they are conformal coated and placed in their enclosures saves considerable time and expense.

Above and beyond testing - ISP/FLASH Memory Programming

In addition to testing boards, your in-circuit tester can also be put to other beneficial uses on the manufacturing floor. An example is the in-system programming of flash devices. Flash devices and In-System Programmable (ISP) devices have the following common characteristics:

- They can be programmed or erased after they have been assembled on a board.
- They remain programmed after power is removed.

The tester uses its Deep Serial Memory (DSM) option to store the firmware data that will be written into the device. This allows you to change the firmware data without having to modify the test program.

In addition to automatically programming flash devices with the correct data, using in-circuit testers for programming flash devices has several other benefits which lower production costs:

- Elimination of production steps
- Elimination of off-line programming equipment
- Reduction in handling damage
- Elimination of programmed parts inventory

So, in-circuit testers can do more than you thought

As you've seen in this chapter, the capabilities of a standard in-circuit tester can be easily expanded. Simply, add a few more test instruments and let those instruments be used either by themselves or closely coupled to the digital subsystem in the tester. The advantages of this arrangement are many:

- The test instruments are all conveniently contained within the tester itself, and they are all controlled by the tester's standard test language.
- Even when operating independently of the digital subsystem, these additional test instruments can provide a lot of testing power. In this chapter alone, you've seen how they can be used to measure signal frequencies, pulse widths, analog voltages, and even perform some dynamic testing of devices.
- When closely coupled to the digital subsystem, these test instruments can be precisely timed to digital inputs. This makes possible high-speed, dynamic testing of both digital and analog signals at the same time. Combinational testing becomes available on an in-circuit tester.
- For measuring critical high-frequency signals, a high-frequency scanner is available to provide a high-fidelity path between the UUT and the tester instrument.

Glossary

Glossary

Analog harmonic testing. A vectorless test method for locating opens and marginal solder connections using the harmonic signatures of an integrated circuit device.

Analog instrument bus. Common lines in which any analog test instrument can be connected through the multiplexer and any UUT circuit can be connected through the scanner.

Analog instrument multiplexer. A program-controlled relay matrix used to connect any analog test instrument to the instrument bus.

Analog tests. Test which measures the values of the analog components on the PCB. See page 9.

ASCII. (American Standard Code for Information Interchange). A standard representation for encoding alphanumeric and special characters as binary values.

ASICs. Application-specific integrated circuit, a semi-custom designed IC.

ATE (Automatic Test Equipment). A system, typically computer-controlled, used to test electronic devices, PCBs and products.

ATG (Automatic Test Generation). Computer generation of a test program based solely on the circuit topology, requiring little or no manual programming effort.

Backdriving. The process of forcing the output of a digital device to a selected logic level.

Backplane. Circuit board at the rear of a crate which, by means of its attached connectors, mates with the modules and constitutes the crates segment.

Bareboards. A bare PCB containing the holes and tracks for mounting and connecting components.

Bed-of-nails fixture. A type of vacuum-activated test fixture in which the nodes on a PCB come in contact with the system test nails.

Boundary scan. The term given one of a variety of testability methods whereby component pin data can be controlled or observed using a serial shift register.

Burst. A series of instructions for testing a particular digital device; executed at high speed under the control of the driver/sensor controller.

BusBust™ feature. A GenRad testing technique used to isolate the device(s) causing a bus failure.

Bus. A signal line or a set of lines used by an interface system to connect a number of devices and to transfer information.

Bus cycle. The process whereby digital signals effect the transfer of data bytes or work across the interface by means of an interlocked sequence of control signals.

Bused devices. Several devices tied to a common bus.

Bus node. A circuit node that can be driven by more than one device.

CAD (Computer Aided Design). The use of computers to aid in design layout and analysis.

Channel. A single path for transmitting electric signals, usually in distinction from other parallel paths.

Circuit Analyzer. The combination in a single enclosure of a plurality of instruments or instrument circuits for use in measuring two or more electrical quantities in a circuit.

Circuit Description. Circuit board information supplied by the user to the tester, allowing the tester to characterize the circuit board to be tested.

Clock drive nails. Nails used to supply programmable clock inputs to the UUT.

Clock sync nails. Nails used to handle test procedures that require each digital test step to be synchronized with a clock on the UUT. The selected sync signal controls the internal timing of the tester during the burst.

Clock synchronization. The high-speed clock module can be synchronized to a signal on the UUT.

Compiler. See Translator.

Contact resistance. The resistance to current flow between two touching materials which is typically created by mechanical connections within the test system.

Converters. Item used to adapt an existing test fixture to a different mechanical receiver interface.

CRC. Cyclic redundancy check.

Data logging software. Software used to accumulate data on board failures for future analysis.

Debug. In testing, the process of modifying a test set to get measured results from a known good board to agree with expected results from that board.

Design-for-testability. Ongoing effort by both component and board designers to improve the observation and control of their designs during test.

Device-under test (DUT). One of the components undergoing in-circuit tests. Compare with UUT.

Digital multiplexing. The sharing of a few common driver/sensors by a group of test nails (e.g., 2 driver/sensors shared by 8 nails).

Digital tests. Test that checks the operation of digital ICs on the PCB. See page 21.

Drivers. The tester circuitry used to force selected logic levels to digital devices on the UUT.

Driver/Sensors (D/S). The tester circuits used to force logic level inputs and monitor logic level outputs from digital devices on the UUT.

Driver/sensor multiplexing. Sharing of driver/sensors among test nails. This gives more pins per driver/sensor and expands pin capability of the system.

Driver strobe. Defines the precise time at which an input test pattern is applied to the UUT.

D/S Controller. The tester hardware module that controls the testing sequence during a digital burst.

Dual-inline-packages (DIP). A component package with two parallel rows of leads intended to allow mounting of ICs.

Dual-well fixtures. A fixture that has two separate bed-of-nails (wells) and two separate sets of controls. While one board is tested, a second board can be loaded. The fixtures are used to speed-up testing in a high-volume test environment.

Fail. In testing, a term applied to a UUT or a device which has one or more failing test steps.

Fault. Any condition that causes a device or circuit to fail to operate in a proper manner.

Fast nails. Test nails that connect to the driver/sensors, and or the analog instrument bus. Can execute up to five million test steps per second.

Fault coverage. A quality grade showing what percentage of conditions causing a device or circuit to fail are not tested.

Feedback loop. A circuit path that causes a device input to be affected by other devices further along the path.

Fixture. The device that interfaces between the tester and the UUT. See Bed-of-Nails fixture.

Fixture nails. The probes in the fixture that contact the UUT.

Flagspecs. Special parameters and instructions supplied by the user to the ATG task.

Flash devices. A special type of EEPROM that can be erased and reprogrammed in blocks instead of one byte at a time. Many modern PCs have their BIOS stored on a flash memory chip so that it can easily be updated if necessary.

Forward bias. The bias voltage which tends to produce current flow in the forward direction.

Functional tester. ATE which tests a UUT as a complete, functional entity, typically by applying inputs and sensing outputs only through the UUT's edge connector.

Gerber data. 'Vector' data derived from the PCB's CAD database which provides layout specifications for the PCB design such as track width and components' X-Y coordinates.

Glitch. A small spurious pulse or spike on a signal line.

GO/NOGO test. A testing process which yields only a pass or fail condition.

Good Board. A fault-free board.

GPIB (General Purpose Instrument Bus). Provides system integrators with a standard electrical and mechanical interface between instruments and PC controllers

Greenlighting. The pass condition achieved upon the completion of the debug process, or sometimes applied to any UUT which passes a test.

Ground. A conducting connection by which an electrical circuit is connected to the earth, or to some conducting body of relatively large extent.

Ground plane. A metallized plate in the fixture which serves as a common ground point to reduce any noise generated by cross-coupling of test signals.

Guarding. In in-circuit testing, the process of ensuring that a shunt path does not interface with testing of the device.

Guard node (or terminal). The common node that connects shunt paths around the unit-under-test to virtual ground isolating the component for accurate test measurements.

High impedance state. See Tri-state.

Hybrid tests. Tests that check components that perform both analog and digital functions.

IEEE-488 Bus. A data transmission bus which provides communication between the tester and external devices.

In-circuit testing. Test method which isolates the component to be tested from all others and performs analog or digital test routines which are automatically generated from a data library. Most commonly used to detect manufacturing or process faults.

In-circuit tester. ATE which tests each separate device on a board by applying test signals directly to the device's inputs and sensing the results directly from the device's output.

Inhibit. If a digital IC cannot be disabled, its effects can be inhibited by placing it in that state in which it can be backdriven with the least stress upon it during the test.

Input vector. A set of logic values to be applied to the complete set of input test nails at any one point in time.

Instrument multiplexer. A set of relays connecting the multiple channels of a tester to the analog instruments.

Integrated circuit (IC). An array of interconnected circuits integrated into a signal chip.

Interstrobe Time (IST). The time between a driver strobe and a sensor strobe.

Inter-Test Time (ITT). The time between two successive driver strobes.

ISP devices. In system programming devices.

Knee voltage test. A test which approximates the exponential characteristics of a diode operating in its forward bias region.

Known good board. A circuit board which is verified to be fault-free.

Land. A portion of a conductive pattern on the PCB used for the connection, or attachment, or both of surface mount components. Also known as a ”pad”.

Lead. Refers to the connection attached to a circuit component.

Learning. The process of determining the expected outputs from a device by applying inputs to a known good board and measuring the actual outputs.

Locating pins. Pins found on the tester’s receiver that fit into the locating holes in the fixture to ensure that the fixture and receiver are properly aligned.

Logic levels. Voltage levels representing a logic 0 and a logic 1.

Looping. The repeated execution of a sequence of statements.

Memory bank. A bank of memory cells attached to each driver/sensor.

Memory cell. A storage cell able to hold one byte of information.

Multiple level digital isolation (MLDI). Applying signals upstream from the device-under-test to control the inputs to the IC and completely isolate the component.

Multiplexing. In a tester, multiplexing refers to the sharing of test instruments among test nails to reduce the size and cost of the system.

Nails. Spring-loaded metal probes used in a bed-of-nails test fixture to make electrical contact with the nodes on a circuit board.

Nail-assignment. The replacement of temporary nail numbers assigned in a circuit description with the target nail numbers used for the actual wiring of the fixture.

Nand tree testing. Using a circuit’s internal NAND tree, in-circuit tests can be developed to check for open input and bi-directional pins, as well as stuck-at input pins.

Node. The electrical interconnection between two or more device leads.

Open. A fault which causes 2 electrically connected points to become separated.

Output Vector. The set of logic values, either expected or measured, for all output pins of a UUT or device at a particular test step.

Pads. See lands.

Panel-test. A test which allows multiple boards to be manufactured on a single panel and tested sequentially without any further operator intervention. The system software reports the test results for each board separately.

Pass. In testing, a term describing a device or a UUT which has no failing test steps.

Poor-wetting. A soldering defect caused by board contamination, solderability problems, or a poor soldering process.

Probe. A device used as a movable test nail to monitor various nodes on the UUT.

Programmable power supply. A power supply capable of having its voltage(s) programmed or selected over a range of values.

Programming station. A test system which does not contain any test hardware, typically used for test program development.

Receiver. That part of the interface between the tester and the fixture that is permanently attached to the tester.

Reflow soldering. A bonding method in which a solder paste is applied to the surfaces of a PCB where components are being attached to the board.

RS-232. A serial interface standard for communicating with byte-oriented equipment, such as teletypes.

Run-Time Controller. System hardware module which handles all data transfers between the CPU and the other modules in the digital subsystem.

Run-Time System. The collection of software programs required to perform the actual testing and diagnosis of a UUT.

Scanner. A program-controlled relay matrix used for connecting any UUT circuit node to the analog instrument bus.

Scan path. The circuit that interconnects a set of boundary scan components through their TDI and TDO pins. Test data and instructions are serially shifted through the scan path.

Schematics. The set of drawings which show the elements of a circuit and how they are interconnected.

SCRATCHPROBING technique. A GenRad diagnostic feature that directs the user to lightly move (scratch) the probe along the pins of an IC to check for continuity.

Sensor strobe. Defines the precise time at which output responses from the UUT are measured.

Short. A fault which causes two or more normally electrically separated points to become connected.

Shorting plate. A conductive plate used to test for opens across the test nails.

Shunt path. A parallel circuit path that takes some of the current away from the desired device.

Sink. Typically refers to current flowing into a power source.

Solder paste. A method of simultaneous flux and alloy application to the soldered joint.

Solder paste stencil. A thin stainless-steel sheet with a circuit pattern cut and etched into the material.

Source. Typically refers to current flowing out of a power source.

Strobes. Several internal pulses from the digital Driver/Sensor system to synchronize external equipment to system timing during debugging of programs or for special testing requirements.

Stuck state. A digital device stuck at 0 or 1 state in the digital testing.

Surface-mount device (SMD). A device soldered directly to the surface of a PCB rather than soldered into holes.

Target nail. Actual nail numbers assigned by the nail assignment software to replace the temporary nail numbers assigned by the user in the circuit description.

Temporary nail. Arbitrary nail numbers assigned by the user when developing the circuit description file.

Test Language. High-level language used to write the test program.

Test nail. See Nails.

Test program. The set of instructions required by the tester which controls the testing of the UUT.

Test program statements. The instruction code within the test program required by the test system for testing the UUT.

Test set. The unique combination of a test program and a test fixture used to test a particular PCB.

Test step. The application of a single input/vector.

Test vectors. A set of logic values to be applied to the complete set of test nails at any one point in time.

Through-hole. A hole in the PCB where a lead from a device goes through the PCB and gets soldered to the other side of the board. Compare with VIA HOLE.

Tombstoning. A picturesque term describing a chip that has worked its way into a vertical position leaving only one end soldered.

Tracks. Paths on the PCB that allow signals to be sent.

Translator. A software module which analyzes and converts the test program from high-level test language to binary machine code. The input to the translator is a source file; the output is an object-code file.

Tri-state (or 3-state). In addition to a high and low state, some devices such as bus drivers have a third (high impedance) state. This third state effectively disconnects the device output from all other circuitry on the board.

Truth table. A table showing the expected outputs from a digital device for all possible combinations of logic inputs to that device.

TTL (Transistor-Transistor-Logic). A logic circuit containing two transistors, for driving large output capacitance at high speeds.

UUT(Unit-Under-Test). Term applied to any circuit board that is being tested by ATE. Compare with DUT.

Vector. See Input vector or Output vector.

Vectorless testing. Technique which detects defective solder joints by measuring the coupling of a signal through the component lead frame to a detector plate or another system pin.

Via hole. A plated hole in the PCB that allows signals on a track to pass from the top of the PCB to the bottom (or from bottom to the top). This hole is not used to hold a lead from a device. Compare with Through-hole.

Virtual ground. A zero voltage potential without being tied directing to ground.

VXIbus (VMEbus Extensions for Instrumentation). Intended to provide an industry standard based on the 32-bit VME data bus with additional high performance instrumentation buses for precision timing and synchronization. This standard allows designers to develop compatible modular instruments.

Wave soldering. A bonding method which involves passing the board over a continuous wave of molten solder, thus soldering all the joints simultaneously.

X-ray testing. A test technique using X-ray equipment to check solder joints.

Focus on In-Circuit Testing

Corporate Headquarters
7 Technology Park Drive
Westford, MA 01886-0033

1-800-4-GENRAD
www.genrad.com

GenRad®
The Technology of Knowledge